CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION

A lot of research has been moving towards future wireless communication standards and systems. During the last decade reconfigurable architectures became the mainstream implementation technology for custom computation and embedded system products in such fields as wireless communication, image processing, video processing, multimedia etc. A significant number of reconfigurable architectures have already been proposed. A lot of work on efficient implementation of wireless networks and cellular communication standards on reconfigurable architectures has been undertaken to date. This chapter gives an overview of such reconfigurable architectures used for various applications.

2.2 RECONFIGURABLE SYSTEM

Implementing a system with an Application Specific Integrated Circuit (ASIC) offers the most optimized solution for a specific application. However, the trend in communication systems is towards more rapidly changing specifications with a shorter time interval between updates of existing standards. Resultantly, communication systems should have a flexible architecture for easy and quick updates of many diverse standards. The ASIC design approach is not suitable for this purpose. Programmable devices such as General Purpose Processors (GPPs) and Digital Signal Processors (DSPs) address flexibility problems. However, such

programmable devices often fail to meet the speed requirements of some communication standards. Further high power consumption and possibly the high cost of these devices make them even less attractive. So Reconfigurable systems are an intermediate approach between the ASICs and GPPs.

2.2.1 Reconfigurability

Reconfigurability is the ability of a device to change its internal structure, functionality and behaviour, either on command or autonomously. Reconfigurable Computing originated in practical terms in the late 1980's with the emergence of FPGAs, which are Integrated Circuits (ICs) whose hardware personality could be completely re-defined simply by loading a new "configuration," just as new software modules can be loaded onto a microprocessor or DSP (Maya Gokhale et al 2005).

2.2.2 Classes of Reconfiguration

Reconfiguration can be classified into static and dynamic reconfiguration.

2.2.2.1 Static reconfiguration

Static reconfiguration is referred to as compile-time reconfiguration, which is the simplest and most common approach for implementing applications with reconfigurable logic. Static reconfiguration involves hardware changes at a relatively slow rate, takes hours, days, or weeks. In order to reconfigure such a system, it has to be halted while the reconfiguration is in progress and then restarted with the new program.

2.2.2.2 Dynamic reconfiguration

Dynamic reconfiguration is run-time reconfiguration, which uses dynamic allocation scheme that re-allocates hardware at run-time. It increases

system performance by using highly-optimized circuits that are loaded and unloaded dynamically during the operation of the system. Dynamic reconfiguration is based on the concept of virtual hardware, which is similar to the idea of virtual memory. In this case, the physical hardware is much smaller than the sum of the resources required by all the configurations. Therefore, instead of reducing the number of configurations that are mapped, it is preferable to swap them in and out of the actual hardware, as they are needed.

2.2.3 Architectural Characteristics

2.2.3.1 Interconnection network

With an increasing number of computational blocks available on a chip, a major bottleneck is caused by the communication among these functional blocks. This creates the need for scalable on-chip communication mechanism. Simple on-chip communication techniques such as buses do not scale when the number of PEs increase. An interconnection network is a programmable system for transporting information between nodes. Interconnection Network consists of three basic components. They are network adapter, router and link. The network adapter provides the separation between the processing core and the communication network. The router implements the routing mechanisms to route the information on the links, which might consist of one or more physical or logical channels. The most fundamental characteristic of an interconnection network is the Network topology which is the structure or organization of the switching and routing nodes and the channels represent the network topology. Examples of network topologies include ring networks, mesh networks, torus networks and a combination of these topologies.

2.2.3.2 Granularity

The granularity of a reconfigurable fabric is defined as the size of the smallest functional unit based on its data width and computational capability. Based on granularity reconfigurable architectures can be broadly divided into two categories, fine-grained and coarse-grained architectures (Todman et al 2005).

2.3 RECONFIGURABLE PROCESSING FABRIC ARCHITECTURES

2.3.1 Fine-Grained Reconfigurable Architectures

Fine-grained reconfigurable architectures consist of PEs and interconnections that are configured at bit-level. Fine-grained systems provide high flexibility and can be used to implement any digital circuit. But, due to fine-grain configuration, these systems exhibit low/medium performance, high configuration overhead and poor area utilization when they are used to implement processing units and datapaths that perform word-level data processing. The most common kind of fine-grained functional units are the small LUTs that are used to implement the bulk of the logic in commercial FPGAs.

2.3.1.1Reprogrammable FPGAs

Reprogrammable FPGAs generally use Static Random Access Memory (SRAM) to store configuration data. The programming architecture of SRAM-based FPGAs is an important factor in the use of FPGAs for reconfigurable computing. Many approaches to programming architecture have been developed for FPGAs. Programming bandwidth, the granularity of the accesses, on-line programmability and the ability to read out the programming data are the main characteristics of programmable FPGAs as reconfigurable computing. Programming bandwidth is the rate at which

configuration data can be sent to the FPGA. The granularity of the FPGA's programmability determines how many resources are configured with the smallest block of programming data. As far as the FPGA chip designer is concerned, this granularity has a significant effect on the cost of the internal logic used to perform the programming. With respect to reconfigurable computing, the main interest in the granularity of FPGA programming data is related to another aspect of an FPGA's programming architecture that is the support provided by FPGAs to perform on-line programming or dynamic configuration of a portion of their logic. This allows the FPGA users to modify the circuit as it executes. The programming granularity, then, has an impact on the smallest amount of data and logic that can be affected when making small changes to a circuit (for instance, reloading the ROMs holding coefficients used in some arithmetic or DSP algorithm). As an example, the Xilinx Virtex series of FPGAs (Virtex, Virtex-E, Virtex-II, Virtex-II Pro, Virtex-4, Virtex-5 and Virtex-6) allow partial reconfiguration of the FPGA.

The smallest amount of data that can be programmed is called a frame, which is typically hundreds to thousands of bits. So, to partially reconfigure the FPGA logic at run time, an entire frame of configuration data must be rewritten to the FPGA for the smallest change. The ability to read FPGA's programming data and other information through the configuration interface has also had an impact on how FPGAs are used in reconfigurable computing. For instance, with the Xilinx Virtex FPGA, the outputs of the slices and the outputs of the Input Output Blocks (IOBs) (to the outside world and to the rest of the FPGA) can be sampled and read out through the FPGA's configuration interface along with the contents of the various RAMs. This can be used as a means of communication between FPGA design and an external host and it has been used for helping designers to debug reconfigurable computing applications.

These fine-grained devices work well for implementing combinational or sequential logic. However, DSP algorithms use arithmetic operations such as multiplication extensively. Mapping a multiplier onto a fine-grained device produces a complex structure that yields poor performance, so some designs actually embed dedicated multipliers into the array. Recently, researches have proposed new classes of reconfigurable devices that incorporate adders, multipliers, LUTs, and other functional units within the cells.

Despite their increasing popularity, FPGAs are still hampered by a few drawbacks when utilized for signal processing applications (Claudio Brunelli et al 2010),

(a) FPGAs are usually configured by means of VHDL or VERILOG programs, or other Hardware Description Languages (HDLs). HDLs are typically bit-oriented and event-driven languages, thus they are not the most convenient way of implementing signal processing algorithms. Indeed, algorithm and application developers have typically a software-oriented background. On the other hand, even though the DSP community is actually strongly investing in the formation of new application engineers with mixed HW/SW curricula, HDLs are intrinsically not able to offer the design productivity needed by today's algorithmic exploration and deployment.

(b) FPGAs are very costly in terms of area occupation and more importantly in terms of power consumption. It is not possible to utilize them as ''soft-ASICs" by just migrating ASIC design style over a different technology support. The costs related to the usage of FPGAs can be justified only when an

optimal utilization of the available resources can be guaranteed. One possibility to do so is the ability to multiplex over time as many different algorithms as possible on the same FPGA logic.

2.3.2 Coarse-Grained Reconfigurable Architectures

Coarse-grained reconfigurable architectures (CGRAs) consist of reconfigurable PEs that implement word-level operations and special-purpose interconnections retaining enough flexibility for mapping different applications onto the system. In these systems the reconfiguration of PEs and interconnections is performed at word-level. Due to their coarse-grain granularity, coarse-grained reconfigurable systems offer higher performance, reduced reconfiguration overhead, better area utilization and lower power consumption than the fine-grained ones. Coarse-grained architecture, on the other hand, is typically much larger, and may consist of ALUs and possibly even a significant amount of storage.

The configurability of fine-grained reconfigurable logic devices allows designers to specialize their hardware down to the bit level, meaning that, if an application requires 8- or 16-bit for an arithmetic operation, the hardware can directly accommodate this requirement. The configurability makes devices, like FPGAs, suitable to implement a large variety of functions directly, and this comes at a significant cost in terms of circuit area, power and speed. Every level of configurability requires more multiplexing, buffering, routing and memory, thus, requiring more transistors and their interconnection. After recognizing these costs, many researchers have proposed more CGRAs as the basis for reconfigurable computing machines.

2.4 A SURVEY ON EXISTING CGRAs

Zhi Alex Ye et al (2000) described the CGRA called Chimaera architecture as shown in Figure 2.1, which tightly couples a processor and a RFU. This RFU is a small and fast FPGA like device, which is structured as a Reconfigurable array of PEs to implement application specific operations. In Chimaera, the RFU is capable of performing computations that use up to 9 input registers and produce a single register result. The RFU is tightly integrated with the processor core to allow fast operation (in contrast to typical FPGAs which are build as discrete components and that are relatively slow).

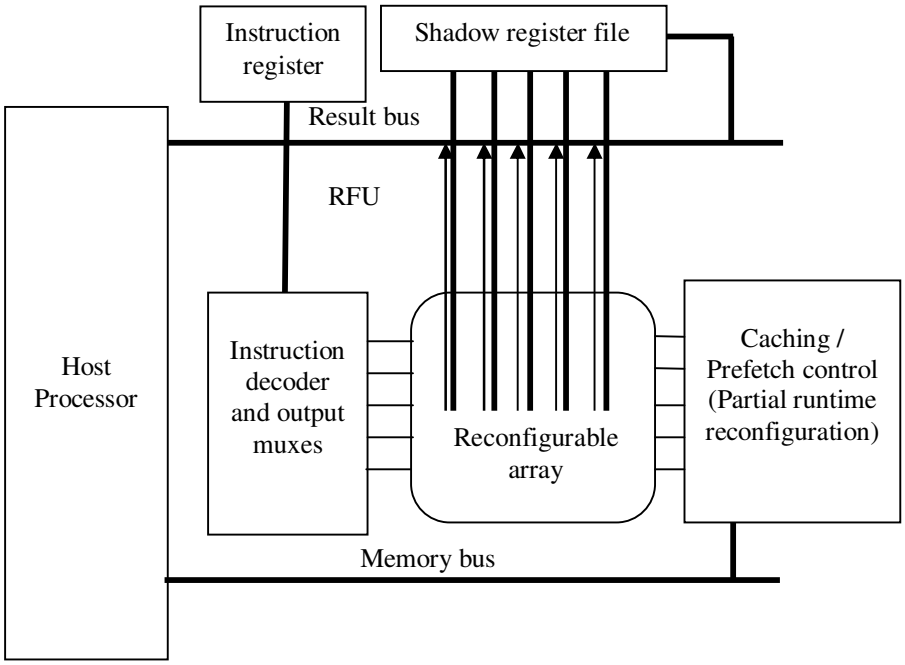

Figure 2.1 Block diagram of chimaera architecture

The RFU can be partially configured so that multiple Virtual Instruction Configurations (VICs) can be cached in it at any given time.

The Chimaera architecture has the following potential advantages

1. The RFU may reduce the execution time of dependent operations. By tailoring its data-path for specific operations, the RFU performs several dependent operations in less time than it takes to execute each of the operations individually.

2. The RFU may reduce dynamic branch count by collapsing code containing control flow into an RFU operation. In this case the RFU speculatively executes all branch paths and internally selects the appropriate one.

3. The RFU may exploit sub-word parallelism. Using the bit-level flexibility of the RFU, several sub-word operations can be performed in parallel.

4. The RFU may reduce resource contention as several instructions are replaced by a single one.

The MorphoSys proposed by Singh et al (2000) comprises a control processor (TinyRISC), a frame buffer (data buffer), a Direct Memory Access (DMA) controller, a context memory (configuration memory) and an array of 64 Reconfigurable Cells (RCs) as shown in Figure 2.2.

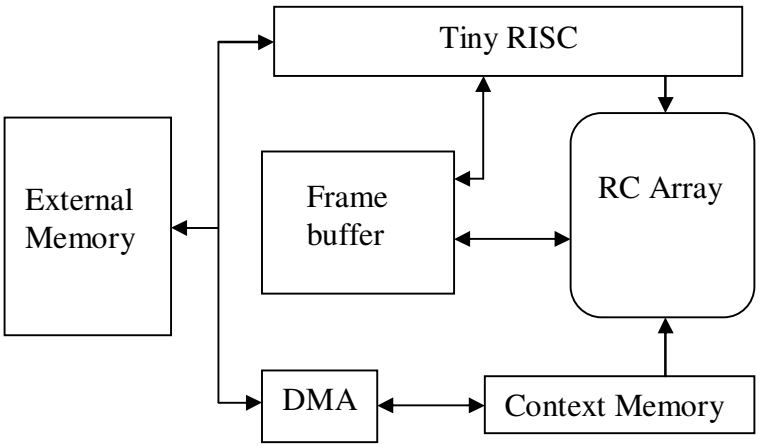

Figure 2.2 MorphoSys architecture

Each RC comprises an ALU-multiplier, a shift unit and two multiplexers at the inputs. Each RC also has an output register, a feedback register and a register file. A context word, loaded from the configuration memory and stored in the context register, defines the functionality of RC.

The potential to integrate application-tailored coarse-grained dynamically reconfigurable architectures into System on Chip (SoC) solutions for future generation mobile terminals has been described by Jurgen Becker and Manfred Glesner (2001). The authors proposed an architecture called DReAM consists of an array of parallel operating coarse-grained Reconfigurable Processing Units (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. In addition, the architecture also provides efficient and fast dynamic reconfiguration possibilities for the RPUs as well as for the interconnection structures.

A reconfigurable computing block targeting handheld devices for 3G communication systems has been suggested by Havinga et al (2001). It can handle both bit-level and word-level operations using Functional Units (FUs) and the resource utilization is low for a mixture of bit-level and word-level operations.

CGRAs have been mainly proposed for accelerating loop structures of multimedia and DSP applications in embedded systems. The more regular structures within the Processing Elements (PEs) with their wider data bit-widths (like 16-bit Arithmetic and Logic Units (ALUs)) and the regularity of the interconnect network between the PEs, greatly reduces the execution time, area, power consumption and reconfiguration time relative to an FPGA device at the expense of flexibility. An automated partitioning methodology between

the heterogeneous Reconfigurable Functional Units (RFUs) such as the fine and coarse-grained reconfigurable hardware of an embedded platform has been introduced by Michalis et al (2006).

Sedcole et al (2006) have proposed two dynamic reconfiguration methods for modular systems in CGRAs. The first method uses partial bit-streams directly to reconfigure the CGRA. However, owing to the organization of configuration memory, in CGRAs, modules exclusively occupy complete vertical sections of the device, thereby severely restricting resource allocation and connectivity. These restrictions are avoided with the second novel merge dynamic reconfiguration method. In this method, information in the partial bit-stream is merged with the current configuration as read back from the device. By using an exclusive-OR function to combine two bit-streams, existing configuration information is preserved and the module can be removed by repeating the XOR operation.

Flexible radio architectures that can support not only multiple standards but also upcoming ones become an important area of research. For the digital part of the radio, domain-specific reconfigurable processors offer the advantage of flexibility as general-purpose processors, and low power by exploiting parallelism in baseband algorithms and providing a direct spatial mapping from algorithms to the architecture, hence, reducing the memory and control overhead associated with general-purpose processors. Most existing techniques have focused on the computation model of the reconfigurable processor, such as PADDI (Chen et al 1992), MATRIX (Mirsky et al 1996), RAW (Waingold et al 2004), and RaPiD (Ebeling 2004). In general, all are composed of an array of heterogeneous coarse-grained reconfigurable units controlled by either Reduced Instruction Set Computer (RISC), or Very Long Instruction Word (VLIW) processor. The granularity of configurable units is usually a word-level operation such as a multiplier, an ALU, or a register. As

the granularity of configurable units directly impacts the energy efficiency of the hardware, Ada (2007) has focused on increasing the granularity of the configurable units without compromising flexibility.

In the book (Stamatis Vassiliadis and Dimitrios Soudris 2007) the authors have briefed about reconfiguration system and an extensive survey of the existing CGRAs from both academia and industry, which indicated both the strengths and limitations of coarse-grained reconfigurable hardware. An important consideration in dynamically reconfigurable systems is the reconfiguration latency and power consumption. Various techniques employed to reduce reconfiguration latency, such as prefetching and configuration caching, have been discussed in this book.

In order to accommodate the variety of reconfiguration processes required by different applications, a tiered framework has been designed by Heng and Ronald (2008). This framework provides standalone reconfiguration capability on the CGRA device as well as a bi-directional communication channel with the embedded host computer to carry out the partial reconfiguration process and routing without manual intervention. It has the ability to generate and reconfigure task bit-streams at runtime as well as design time.

The reconfigurable architecture used by Nazish Aslam et al (2008) has offered a very high number of parallel independent processing units running concurrently. All the individual units are expected to be approximately configured at each execution cycle, where this configuration is performed by a large configuration stream, or a wide instruction, fetched from the program memory.

Resource utilization is a key factor to achieve high performance in reconfigurable architectures and it can serve as a key metric to decide an

appropriate architecture type. Jong-Suk Lee and Dong Sam Ha (2009) have proposed a new CGRA called FleXilicon, which improves resource utilization and achieves a high degree of Loop Level Parallelism (LLP). To estimate the performance of FleXilicon, five different types of applications commonly used in wireless communication and multimedia applications have been implemented and compared their performance with an ARM processor and a Texas Instruments (TI) DSP. It has been proved that FleXilicon reduces the number of clock cycles and increases the speed for all five applications.

Reducing power is very crucial for CGRAs, which is a more competitive and reliable processing core in embedded systems. Yoonjin Kim et al (2009) have described that power reduction can be achieved by using the characteristics of loop pipelining, which is a Multiple Instruction-stream Multiple Data-stream (MIMD)-style execution model. Since MIMD allows simultaneous execution of multiple iterations of a loop in a pipeline, it utilizes PEs better through loop pipelining.

Future embedded systems will require a higher degree of customization to manage the growing complexity of the applications. At the same time, they must continue to facilitate a high degree of flexibility. Lam and Srikanthan (2009) have presented a framework that enables rapid design exploration for Reduced Instruction Set Processors (RISPs), which incorporates reconfigurable structures that are similar to commercially available technologies (i.e. Look Up Table (LUT) based FPGAs with coarse-grained arithmetic units). In particular, the proposed framework can effectively select custom instructions that maximize area utilization of the reconfigurable space without compromising the performance gain. Increasingly, commercial FPGAs are incorporating coarse-grained functional blocks to facilitate high-speed complex arithmetic operations such as

multiplication and division that cannot be efficiently mapped onto the fine-grained logic blocks.

In the field of CGRAs, two major trends have been reported. They are

(1) Coarse-grained reconfigurable arrays provide operation level reconfigurability at the cost of low utilization of the underlying hardware.

(2) Reconfigurable Arithmetic Units (RAUs) provide only sub-word reconfigurability to arithmetic datapaths when data with lower precision are available.

SotirisXydis et al (2009) have extended the two aforementioned major trends by introducing operation level reconfigurability in arithmetic data-paths which incorporate a significant degree of computation density. Specifically, the proposed techniques incorporate flexibility by mapping together the behaviours of a Carry-Save (CS) multiplier, a CS adder and a CS subtractor onto a stable interconnection scheme. The introduced architecture is a coarse-grained reconfigurable data-path that mainly targets ASIC implementation technologies. It provides fast implementations for the set of mapped operations due to the CS-based logic, which eliminates the time consuming carry propagation.

Claudio Brunelli et al (2010) have described the design and implementation of a coarse-grained reconfigurable machine with sub-word and floating point computation support. As an example of the effectiveness of this solution, the mapping of kernels belonging to H.264 and MP3 decoders, plus a kernel from 3D graphics pipeline have been discussed. The mapping has been tested on a prototype of the device implemented on a FPGA board. The software implementation is a C algorithm running on a 32-bit

programmable microprocessor (Coffee RISC core). These implementations have been tested on FPGA running at 50 MHz, thus the corresponding execution times are 2.7 ms and 0.7 ms respectively, thus the speed-up is almost 4 times.

Mahesh and Vinod (2010) have proposed two architectures that integrate reconfigurability and low complexity to realize Finite Impulse Response (FIR) filters. The architectures proposed by the authors are called Constant Shifts Method (CSM) and Programmable Shifts Method (PSM). The CSM architecture results in faster coefficient multiplication operation at the cost of a few extra adders compared to the PSM architecture whereas the PSM architecture results in fewer number of additions and thus less area and power consumption compared to the CSM architecture.

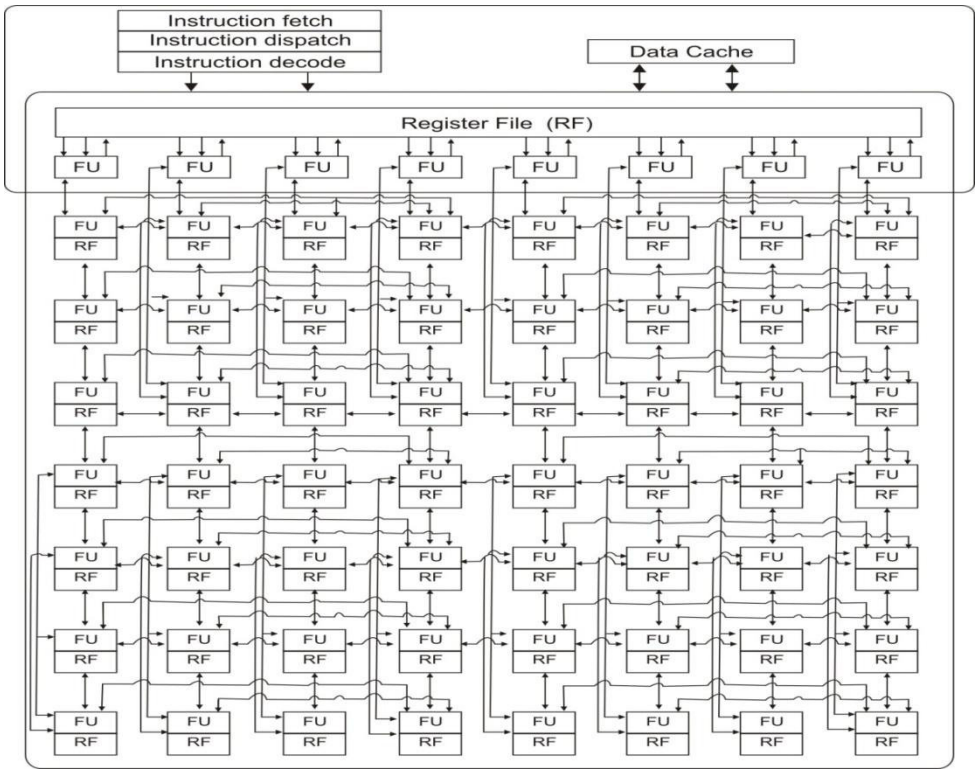

Figure 2.3 Architecture of ADRES

Another example of CGRA is the ADRES architecture (Mei et al 2010) which is shown in Figure 2.3. Each reconfigurable functional unit in this device contains a 32-bit ALU which can be configured to implement one of several functions including addition, multiplication and logic functions, with two small register files. Clearly, such a functional unit is far less flexible than the fine-grained functional units, however, if the application requires functions which match the capabilities of the ALU, these functions can be very efficiently implemented in this architecture.

At the University of Twente, in the Chameleon project (Rauwerda et al 2010), the steps have been made to define an energy efficient heterogeneous reconfigurable SoC architecture. Chameleon systems created one of the first practical commercial implementation of coarse-grained reconfigurable technology for data intensive Internet, DSP and other high performance telecommunication applications (Reiner Hartenstein 2001).

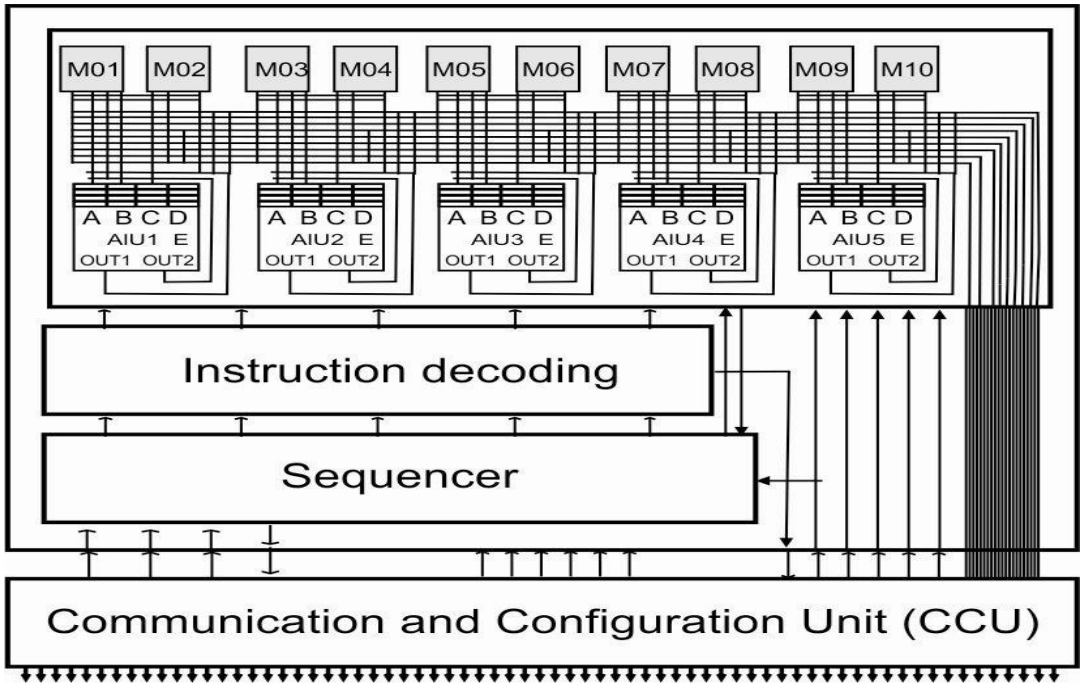

Figure 2.4 MONTIUM processor tile

This architecture contains a coarse-grained reconfigurable part called MONTIUM processor tiles. The MONTIUM tile is designed to execute highly regular computational intensive DSP kernels using it's multiply and accumulate operations. This architecture yields a combination of performance, flexibility and energy-efficiency. Figure 2.4 depicts a single MONTIUM Processor Tile. The hardware organization within a tile is very regular and resembles VLIW architecture. The five identical ALUs (ALU1 to ALU5) in a tile can exploit spatial concurrency to enhance performance. This parallelism demands a very high memory bandwidth, which is obtained by having 10 local memories (M01 to M10) in parallel. The input registers of ALU provide an even more local level of storage. Locality of reference is one of the guiding principles applied to obtain energy efficiency in the MONTIUM.

A vertical segment that contains one ALU together with its associated input register files, a part of the interconnect and two local memories is called a Processing Part (PP). The five PPs together are called the Processing Part Array (PPA). A relatively simple sequencer controls the entire PPA. The Communication and Configuration Unit (CCU) implements the interface with the world outside the tile. The interconnect provides a flexible routing within a tile. The configuration of the interconnect can change every clock cycle. A HiperLAN/2 receiver has been implemented in this architecture, which is adaptable depending on the modulation scheme used, as well as a flexible Bluetooth receiver on the same architecture.

2.5 A SURVEY ON INTERCONNECTS

While reconfigurability provides flexibility required by the applications, high energy consumption of interconnects reminds us that careful optimizations are needed for reconfigurable interconnect structure.

Marlene Wan et al (2001) have suggested three most promising interconnect architectures (Multi-bus, Irregular mesh and Hierarchical mesh).

Multi-bus(IEEE 796): This architecture provides full connection flexibility and has the advantage of simple implementation, but it suffers from a large area overhead (due to the large number of global buses) and high energy consumption (due to the long global buses and the large number of switches).

Irregular mesh: In order to optimize local connections between neighbouring modules, an irregular mesh interconnect structure extended from FPGA is used. Given a module placement, wiring channels are created along the sides of each module and switch boxes are placed at channel intersections. The advantage of a mesh structure is that it can provide low-cost connections between local modules. But long connections between distant modules suffer from a large number of switches in the connection, which results in high energy and delay cost.

Hierarchical mesh: While continuing to exploit the locality of interconnections, adding hierarchies to the interconnect network can reduce the number of switches in long connections.

The increasing integration density of SoCs leads to highly parallel systems with an increasing number of PUs. For scalability reasons the communication architecture has to support several communication paths simultaneously. While traditional communication architectures may fulfil these requirements for a fixed SoC design, changing composition, number and locations of processing modules in runtime reconfigurable SoCs require new communication paradigms. Special communication architectures especially for use in runtime reconfigurable SoC designs have been presented by Thilo Pionteck et al (2008). The issue of runtime reconfiguration also demands

more flexibility of the communication infrastructure. Evaluating the above said interconnect structures, the proposed architecture uses the low cost and flexible interconnect structure which connects each of the functional units to all its surrounding functional units.

2.6 A SURVEY ON WCDMA IMPLEMENTATION

2.6.1 WCDMA Technology

WCDMA supports mobile/portable voice, images, data, and video communications up to 2 Mbps (local area access) or 384 kbps (wide area access). WCDMA is based on Direct Sequence-Code Division Multiple Access (DS-CDMA) technology in which user-information bits are spread over a wide bandwidth (much larger than the information signal bandwidth) by multiplying the user data with the spreading code. The chip (symbol rate) rate of the spreading sequence is 3.84 Mcps for the WCDMA system that needs 5 MHz bandwidth for deployment. The name wideband is derived to differentiate it from the 2G CDMA (IS-95), which has a chip rate of 1.2288 Mcps and lesser bandwidth. In a CDMA system, all users can be active at the same time on the same frequency and are separated from each other with the use of user specific spreading codes. The wide carrier bandwidth of WCDMA allows supporting high user-data rates and also has certain performance benefits by exploiting the multipath diversity. The actual carrier spacing to be used by the operator may vary on a 200-KHz grid between approximately 4.4 and 5 MHz, depending on spectrum arrangement and the interference situation.

2.6.2 Modes of WCDMA

WCDMA supports two basic modes of operation, Frequency Division Duplex (FDD) and Time Division Duplex (TDD). In the FDD mode,

separate 5-MHz carrier frequencies with duplex spacing are used for the uplink and downlink, respectively, whereas in TDD only one 5-MHz carrier is time shared between the uplink and the downlink. The TDD mode was not used in the WCDMA network deployment. WCDMA operates with asynchronous base stations, without the need for chip level timing source. From the signal detection point of view WCDMA uses coherent detection based on the pilot symbols and/or common pilot. WCDMA allows many performance enhancement methods to be used, such as transmit diversity using advanced CDMA receiver.

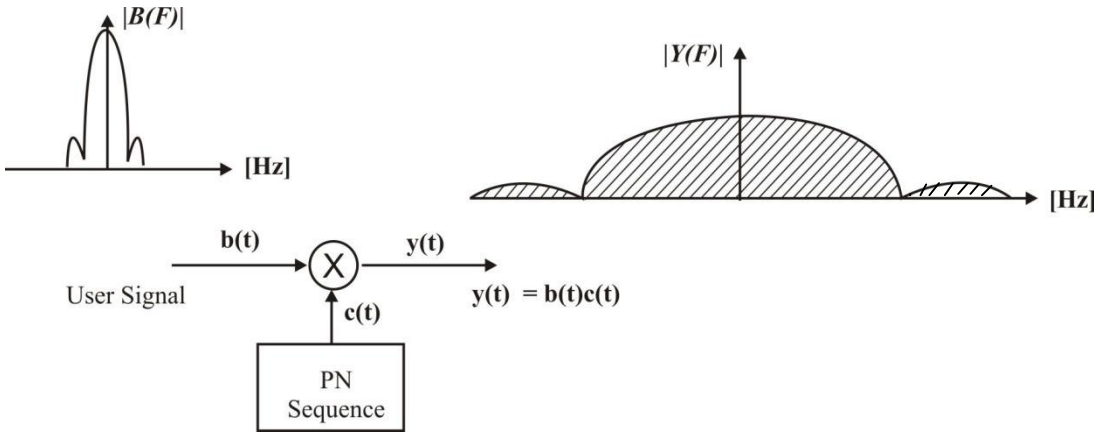

Figure 2.5 Block diagram of DS spread spectrum transmitter

In the block diagram of DS spread spectrum transmitter, as shown in Figure 2.5, the narrowband user information signal b(t) is spectrally spread to become y(t), using the PN code sequence c(t) (Joseph Boccuzzi 2008).

The benefits of WCDMA systems lie in the diversity gains that can be achieved over multipath propagation channels and the significant improvement of intercellular handovers. Also, WCDMA has resistance to narrowband jamming i.e., the wideband signals can be hidden at very low

power spectral density levels, which was of importance in the early military roots of CDMA technology (Rudolf Tanner and Jason Woodard 2004).

2.6.3 Spreading and Scrambling

WCDMA uses a two level code system, orthogonal spreading codes and pseudo random scrambling codes. In order to support variable data rates, the WCDMA air interface allows for per-channel selectable spreading factors, and this family of spreading codes is called OVSF codes. The use of OVSF and scrambling codes is different in the downlink and uplink. In the downlink, the OVSF codes, also called channelization codes, are used to multiplex different channels transmitted in the same cell. In the uplink, the OVSF codes are used to separate data and control channels from a specific user. Scrambling, using pseudorandom sequences, is used in addition to spreading. In the downlink, different scrambling codes separate different cells, and in the uplink they separate different users.

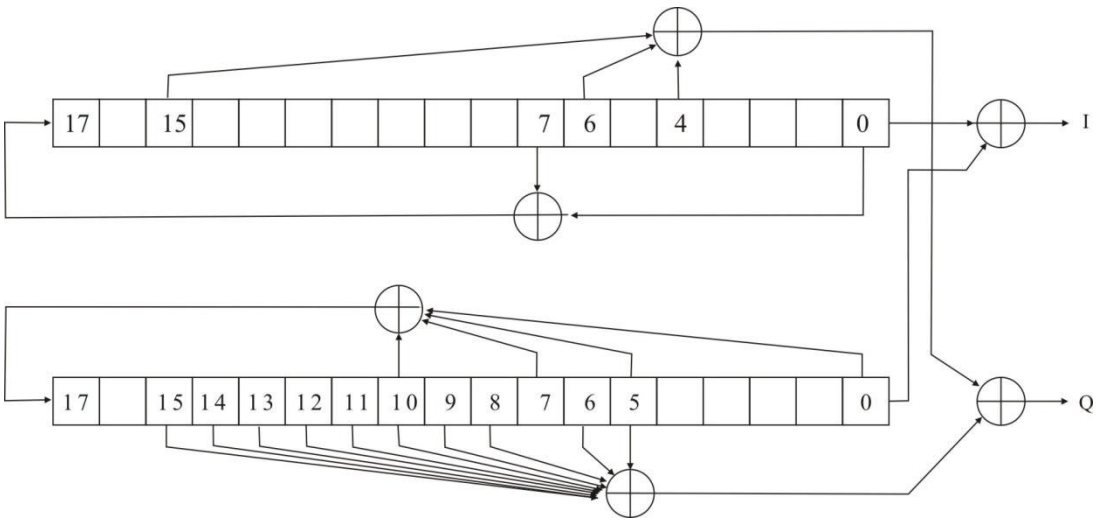

Figure 2.6 Downlink scrambling code generator

In the downlink, the shift register arrangement as shown in Figure 2.6, allows 2^{18}-1 different scrambling codes to be generated. However, for ease of code acquisition and cell search, not all of these are used. The scrambling codes are arranged into 512 sets of primary scrambling codes, each with an associated 15 secondary codes. The 512 primary codes are further divided into 64 code groups of 16 codes. Each code has associated with it an alternative left and right associated scrambling code that can be used in compressed mode.

2.6.4 RAKE Receiver

A RAKE receiver is an integral part of a wireless CDMA communication system, combining the received signals from several multipaths into a common decision statistic with significantly improved Signal-to-Noise Ratio (SNR) and robustness to channel fading. These RAKE receivers are essential while incurring a huge computational complexity. ASIC implementations provide high performance, but are not flexible enough to adapt to the various situations and services in next generation of wireless communication systems. Jurgen Becker and Manfred Glesner (2001) have mapped computation-intensive RAKE receiver component of 3G WCDMA based mobile communication systems onto the DReAM architcture.

Hyung-Jin Lee and Dong Sam Ha (2002) have presented a new architecture for power and area efficient RAKE receivers for 3G wireless WCDMA systems. This architecture is based on parallel operations of code generators and shares Orthogonal Variable Spreading Factor (OVSF) code generators and scrambling code generators among all fingers, which leads to power and area reduction. A reconfigurable architecture for WCDMA mobile applications has been addressed by Ahmad (2002). It is well suited for data path-oriented operations, but homogeneous PEs degrade the resource utilization and complicate the routing and the control.

In the architecture proposed by Zhuan Ye et al (2003), a scalable software approach to implement the RAKE receiver algorithm has been presented. This architecture allows the RAKE receiver for different CDMA standards to be realized using different software on the same hardware.

As mobile devices are battery powered equipments, it is necessary to analyze and reduce power consumption of the proposed architectures. Two architectures for low power flexible RAKE receivers have been presented by Boris et al (2004) based on alternative memory organizations of the input sample buffer block. Both architectures provide more power savings as compared to the conventional RAKE receiver.

Proficient architectures for IS-95A CDMA transceivers have been modelled and all the modules (Convolutional encoder, Cyclic Redundancy Check (CRC) generators, Walsh Code generator, Data burst randomizer, Block interleaver and Viterbi decoder) of IS-95A CDMA have been simulated using Xilinx Virtex V100CS144 FPGA device by Rajaram Sivasubramanian and Abhaikumar Varadhan (2006).

Though the above given architectures on WCDMA are implementing different levels of approach for area and power reduction, more efficient implementation techniques are still needed to achieve better area efficiency compared to the traditional architecture design.

2.7 A SURVEY ON OFDM IMPLEMENTATION

2.7.1 Basic Principles of OFDM

OFDM has been adopted as the downlink transmission scheme for the 3GPP Long-Term Evolution (LTE) and is also used for several other radio technologies, e.g. standards like IEEE 802.11a&g (Wi-Fi) for WLANs, the growing IEEE 802.16 (WiMAX) for Metropolitan Access (Ahmad Sghaier

et al 2008) and Digital Video Broadcast (DVB) technologies. WCDMA multi-carrier evolution to a 20MHz overall transmission bandwidth could consist of four (sub) carriers, each with a bandwidth in the order of 5 MHz. In comparison, OFDM transmission may imply that several hundred subcarriers are transmitted over the same radio link to the same receiver. In complex baseband notation, a basic OFDM signal x(t) during the time interval $mT_u \leq t < (m+1)T_u$ can thus be expressed as

$$x(t) = \sum_{k=0}^{N_c-1} x_k(t) = \sum_{k=0}^{N_c-1} a_k^{\,m} e^{j2\pi k\Delta f t} \tag{2.1}$$

where $x_k(t)$ is the k^{th} modulated subcarrier and $a_k^{\,m}$ is the complex modulation symbol applied to the k^{th} subcarrier during the m^{th} OFDM symbol duration, i.e. during the time interval $mT_u \leq t < (m+1)T_u$. OFDM transmission is thus block based, implying that, during each OFDM symbol duration, N_c modulation symbols are transmitted in parallel. The modulation symbols can be from any modulation alphabet, such as QPSK, 16QAM, or 64QAM. The term Orthogonal Frequency Division Multiplex is due to the fact that any two modulated OFDM subcarriers $x_{k1}(t)$ and $x_{k2}(t)$ are mutually orthogonal over the time interval $mT_u \leq t < (m+1)T_u$, i.e.

$$\int_{mT_u}^{(m+1)T_u} x_{k1}(t)x_{k2}(t)dt = \int_{mT_u}^{(m+1)T_u} a_{k1}a_{k2}^{\,*} e^{j2\pi k_1\Delta f t} e^{-j2\pi k_2\Delta f t} dt = 0 \quad \text{for } k_1 \neq k_2 \tag{2.2}$$

Thus basic OFDM transmission can be seen as the modulation of a set of orthogonal functions $\phi_k(t)$, where

$$\phi_k(t) = \begin{cases} e^{j2\pi k\Delta f t} & ,0 \leq t \leq T_u \\ 0 & ,\text{otherwise} \end{cases} \tag{2.3}$$

2.7.2 OFDM Demodulation

Taking into account the orthogonality between subcarriers according to Equation (2.2), it is clear that, under ideal condition, two OFDM subcarriers do not cause any interference to each other at the receiver. The avoidance of interference between OFDM subcarriers is not simply due to a subcarrier spectrum separation, for example, the case for the kind of straightforward multi-carrier extension like in normal Frequency Division Multiplexing (FDM) with guard bands between carriers. Rather, the subcarrier orthogonality is not due to the specific frequency-domain structure of each subcarrier in combination with the specific choice of a subcarrier spacing f equal to the per-subcarrier symbol rate $1/T_u$. However, this also implies that, in contrast to the kind of multi-carrier transmission outlined, any corruption of the frequency-domain structure of the OFDM subcarriers, e.g. due to the frequency selectivity of the radio channel, the inter-subcarrier orthogonality will be lost, and this leads to interference between subcarriers. To handle this and to make an OFDM signal truly robust to radio-channel frequency selectivity, cyclic-prefix insertion is used.

2.7.3 OFDM Implementation using IFFT/FFT Processing

Due to its specific structure and the selection of a subcarrier spacing f equal to the per-subcarrier symbol rate $1/T_u$, OFDM enables low-complexity and computationally efficient FFT implementation for processing. To verify this, consider a time-discrete (sampled) OFDM signal where it is assumed that the sampling rate f_s is a multiple of the subcarrier spacing f, i.e. $f_s=1/T_s=N\times f$. The parameter N should be chosen so that the sampling theorem criterion is satisfied. As $N_c \times f$ can be seen as the nominal bandwidth of the OFDM signal, this implies that N should exceed N_c with sufficient margin. With these assumptions, the discrete-time OFDM signal can be expressed as

$$x_n = x(nT_s) = \sum_{k=0}^{N_c-1} a_k e^{j2\pi k\Delta f n T_s} = \sum_{k=0}^{N_c-1} a_k e^{j2\pi kn/N} = \sum_{k=0}^{N-1} a_k' e^{j2\pi kn/N}$$

Where

$$a_k' = \begin{cases} a_k & ,0 \le k < N_c \\ 0 & ,N_c \le k < N \end{cases} \qquad (2.4)$$

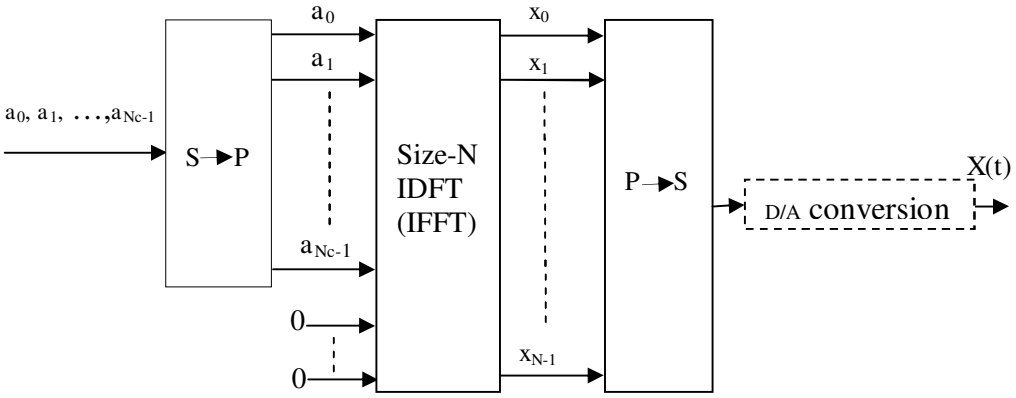

Figure 2.7 OFDM transmitter using IFFT processing

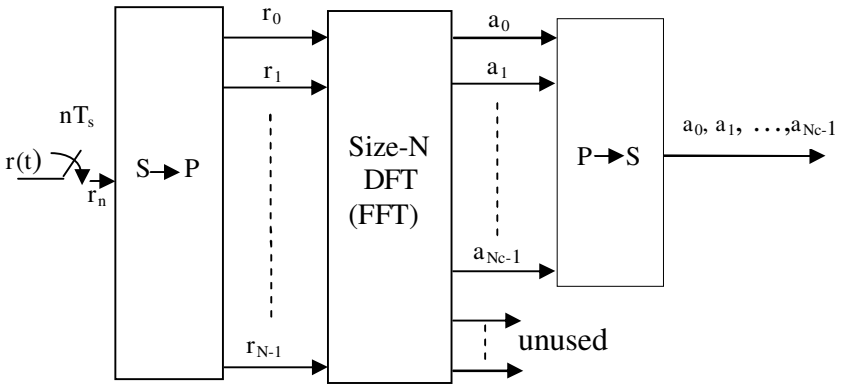

Figure 2.8 OFDM receiver using FFT processing

Thus, the sequence x_n, i.e. the sampled OFDM signal, is the size-N Inverse Discrete Fourier Transform (IDFT) of the block of modulation symbols a_0, a_1, . . . , a_{Nc-1} extended with zeros to length N. An OFDM

transmitter can thus be implemented by means of IDFT processing followed by digital-to-analog conversion, as illustrated in Figure 2.7. Especially, by selecting the IDFT size N equal to 2^m for some integer m, OFDM modulation can be implemented using an efficient radix-2 Inverse Fast Fourier Transform (IFFT) processing. It should be noted that the ratio N/N_c, which is a non-integer number, is the over-sampling ratio of the discrete-time OFDM signal. Similar to the OFDM transmitter, efficient FFT processing can be used for OFDM reception, replacing the bank of N_c parallel demodulators with sampling rate $f_s = 1/T_s$, and this DFT/FFT processing is illustrated Figure 2.8.

The physical layer of both HIPERLAN/2 and IEEE 802.11a systems is based on coded OFDM modulation scheme and are almost identical. The physical layers are multi-rate type allowing control of link capability between access points and mobile terminals according to interference situations and distance. The functional blocks of the physical layers of HIPERLAN/2 and IEEE 802.11a have been simulated by Spyridon Blionas et al (2003).

Park et al (2006) have presented an efficient FPGA design of a Multi Input Multi Output-OFDM (MIMO-OFDM) physical layer. He described the role of dynamic reconfiguration for MIMO-OFDM systems. Such a system can achieve flexibility with regard to changing data rates, increasing range and diversity, while offering efficient resource utilization.

Ahmad Sghaier et al (2008) have used two approaches to translate the IEEE 802.16-2004 OFDM functions into VHDL. The first approach targets a lower level of abstraction, using VHDL for programming and utilizing the Xilinx ISE v8.2 for development. The second approach utilizes AccelDSP, which provides a design environment at a higher level of abstraction. The tool is based on translating a floating-point MATLAB code into an HDL code that can be mapped on an FPGA.

FFT and Inverse FFT (IFFT) are the key kernels of OFDM based WLAN systems. The performance of the FFT computation plays a crucial role on the overall operation of OFDM modems. Efficient implementations for performing FFT computations have been presented by Tze-Yun Sung et al (2010). They proposed 128-point COordinate Rotation DIgital Computer (CORDIC)-based split-radix FFT processor architecture, which can be used as a reusable IP core for various FFT with multiples of 128 points. ROM-free twiddle factor generator to eliminate the overhead of load and storage of twiddle factors has also been discussed by them.

In this section a few efficient design techniques for OFDM functions have been explained. In the radix-2 Decimation in Frequency (DIF) FFT algorithm, received signal is multiplied by the twiddle factor after subtraction. But multiplier requires a large silicon area when implementing it on a reconfigurable architecture. Therefore efficient implementation techniques are proposed in this thesis to reduce the number of multipliers.

2.8 A SURVEY ON MULTISTANDARD ARCHITECTURES

In spite of the good flexibility of reconfigurable architectures, there are no such reconfigurable architectures that can fit all possible applications. Therefore, many reconfigurable architectures dedicated to different applications have been devised and developed. This section discusses about the various reconfigurable architectures suitable for multistandard wireless communication applications.

A new coarse-grained approach has been proposed by Helmschmidt et al (2003) to implement the reconfigurable hardware, which is in the form of an array of PEs and also contains resource management mechanisms. The functionality of a reconfigurable array is defined by software-based configurations, which describe the behaviour of PEs and the routing between

them. Special hardware protocols implemented in the communication and control structures of the array ensure that configurations cannot be overwritten illegally. The authors have described the operation of a Software Defined Radio (SDR) solution for Universal Mobile Telecommunications Systems (UMTS)/WCDMA and modern wireless LAN protocols involving a reconfigurable processing array and a DSP/ microcontroller. A multi-standard, multilink wireless terminal must provide the capability of handling these protocols simultaneously. By time-slicing the processing of both protocols over the same hardware, a large saving in the resources required can be achieved.

Jina and Dong (2005) have proposed a Reconfigurable Modem (RM) architecture targeting 3G multistandard wireless communication system. This architecture targets two 3G wireless standards WCDMA and CDMA 2000. This architecture includes cell searcher, RAKE receiver and Viterbi decoder. The cell searcher requires registers, adders, comparators and multipliers. The RAKE receiver contains registers, counters, logic gates, adders, subtractors, accumulators, multipliers, shifters and comparators. The Viterbi decoder requires registers, adders and comparators. Based on the above four different types, PEs which consists of many functional units are constructed. Each PE is dedicated for one of four basic operations (Bit manipulation, One-Bit Correlation, Multiply and Accumulate or ACS (ADD Compare and Select)). Then PEs are grouped into four different types of PE modules. The PEs inside the PE module communicate with each other through reconfigurable bi-directional data paths. It is proved that this architecture achieves flexibility, scalability and low circuit complexity. It can support multiple standards and easily adapt to rapid changes of specification. But this architecture has a critical path delay of 13nsec and that for the ASIC implementation is 7.5nsec. The longer critical path delay is due to long data path inside the PEs.

In wireless communication systems, radio spectrum, radio access technologies and protocol stacks vary from system to system and network to network. Moreover, the evolution of new standards has not stopped and there are no signs of it in the near future, rather there exist incompatible network technologies. The different types of applications and usages have led to the development of different standards being used in wireless communication systems. Though many of the wireless communication systems have almost the same functional blocks, however the way these blocks function differs greatly from standard to standard. Najam-ul-Islam Muhammad et al (2007) have proposed a global air-interface hardware processing block that can be configured to support almost all the existing and in-progress wireless communication standards. The design is based on identifying small macro-processing blocks which can be reused for multiple standards, thus providing hardware flexibility and high efficiency.

To provide the flexibility for supporting seamless services between various wireless networks, there is a high demand for a common hardware platform that can support multiple protocols implemented or controlled by software, generally referred to a SDR (Timo Vogt and Norbert When 2008). Lu et al (2008) have proposed a novel, heterogeneous, reconfigurable architecture for WLAN baseband processing. It consists of four reconfigurable execution units which are tailored to meet the various kinds of computations required for these applications. These units are connected to each other through a reconfigurable interconnection platform, which provides storage for the execution units. The authors have proved that this architecture is capable of meeting the throughput requirements of the IEEE 802.11a/b/g standards at their highest data rates. This architecture functions effectively as a configurable platform which is flexible enough to support multiple functions and multiple standards in wireless communication systems.

Heterogeneous reconfigurable hardware platforms offer the necessary flexibility for performing multiple wireless communication standards and can achieve the performance required by the wireless standards. The support of multiple wireless communication standards introduces a first level of adaptivity in the wireless terminal because the terminal can switch between wireless communication standards. For example, when packet data transport is performed over UMTS, a WLAN hotspot becomes available, the terminal can switch from UMTS to a WLAN standard. The implementation of a WCDMA and an OFDM receiver using the same coarse-grained reconfigurable MONTIUM tile processor have been discussed by Gerard et al (2008).

Support of multiple radio standards will be a key feature of next-generation mobile receivers. It is challenging to integrate these computationally complex radio standards on a signal processing architecture that is both area and power efficient. Srinivasa Chaitanya et al (2009), have presented a SoC solution for dual-mode WCDMA/OFDM receiver using a CORDIC hardware accelerator. In this method, the most computational intensive tasks of the algorithms are identified and performed on a dedicated hardware accelerator using CORDIC PEs. The aforementioned approaches comprise arrays of computational cells enabling mapping of all the desired behaviours. However, this is achieved at the expense of large hardware area overheads and low utilization of the computational resources.

2.9 A SURVEY ON RESOURCE SHARING

There have been considerable research efforts needed for developing reconfigurable architectures that can share its computational PEs among the applications. Shen-Chuan Tai et al (2003) have implemented a single Butterfly Processing Element (BPE) on a single VLSI chip, which can

efficiently share the OFDM's FFT function and IMDCT (Inverse Modified Discrete Cosine Transform) function.

Lasse Harju and Jari Nurmi (2005) have presented a baseband receiver platform that enables software-defined implementation of WCDMA and OFDM baseband receiver functionalities. The platform is composed of a RISC core and a combination of coprocessors to implement the most critical computation kernels of the receiver algorithms. In this platform computational resources of conventional FFT and RAKE receiver architectures are shared by both receiver architectures. This thesis provides a more efficient resource sharing reconfigurable architecture for WCDMA/OFDM based multistandard system than the architecture presented by them since the proposed architecture reduces large number of computational resources by exploiting efficient algorithmic implementations.

2.10 A SURVEY ON MAPPING APPLICATIONS ON RECONFIGURABLE ARCHITECTURE

Various ways of mapping applications on the reconfigurable architecture have been discussed in the following papers. The Chimaera RFU described by Hauck et al (2004) consists of host microprocessor with a RFU for implementing custom instructions on applications. Through several hand mappings to the RFU, the power efficiency of the Chimaera system has been improved.

Ying-Chang Liang et al (2005) have proposed hardware and software sharing among different types of Cyclic Prefix (CP)-based communication systems, including OFDM, Single Carrier Cyclic Prefix (SCCP) system, Multi Carrier-CDMA (MC-CDMA), MC-Direct Sequence-CDMA (MC-DS-CDMA), CP-based CDMA (CP-CDMA) and CP-based Direct Sequence- CDMA (CP-DS-CDMA). An attempt has been made to map

the functional reconfigurable transceiver onto the proposed hardware platform. By identifying the reusable common blocks, a hardware platform has been proposed to support the reconfigurable architecture. The different functional entities required to perform reconfiguration and realize the transceiver have also been explained in this paper.

Some systems usually combine reconfigurable hardware with a software-programmable general-purpose microprocessor. The microprocessor typically executes non-critical control intensive parts of the applications and provides software programmability. Compute-intensive sections, called kernels, are implemented in reconfigurable hardware. The operation parallelism present in kernels is exploited by the available abundant PEs of the reconfigurable hardware, resulting in performance improvements. This type of hardware consists of PEs with word-level data bit-widths (like 32-bit ALUs) connected with a reconfigurable interconnect network has been explained by Michalis et al (2007). This architecture is more efficient in terms of performance, energy consumption and area.

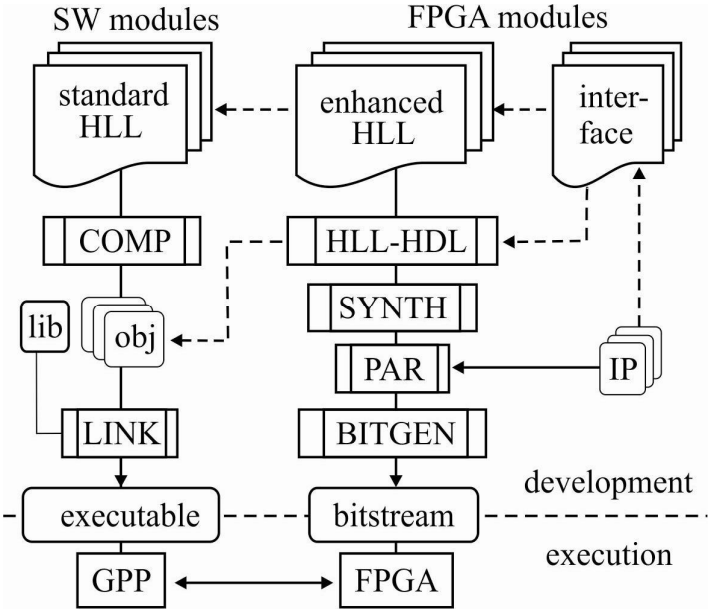

Figure 2.9 Hybrid reconfigurable computing development flow

A hybrid reconfigurable computing development flow is idealized as proposed by Morris and Prasanna (2008), which is shown in Figure 2.9. The application is partitioned into software modules, which are targeted for execution on General Purpose Processors (GPPs), and FPGA modules, which are targeted for execution on FPGAs. The software modules are written in a standard high level language like C or FORTRAN and employ traditional software development tools like compilers and linkers. The FPGA modules are written in an enhanced high level language that allows pipelined loops, parallel code blocks, synchronization primitives, etc. From the perspective of the software module code, the FPGA module is just a parameterized subroutine call rather than a hardware device. An interface mechanism allows the High Level Language-Hardware Descriptive Language (HLL–HDL) compiler to obtain visibility into the Intellectual Property (IP) cores provided by the designer. From the perspective of the FPGA module high level language code, the IP cores are also parameterized subroutine calls rather than hardware devices. High level language compiler produces HDL that is processed by the standard FPGA tool chain, i.e., synthesis, place and route, and bit-stream generation. At run time, the executable is loaded into GPP memory, and the configuration bit-stream is loaded onto the FPGA. The GPP and FPGA-based kernel then cooperatively perform the specified computations.

Dynamically Reconfigurable SoC (RSoC) architectures, which integrate in the same die embedded microprocessors, on-chip memory, reconfigurable logic blocks, and multiple IP cores, are now practical and commercially available (Zexin Pan and Earl Wells 2008). Such architectures promise the flexibility of traditional general-purpose processors while also providing the efficiency and high performance of ASICs. An example is the Xilinx Virtex-4 family of FPGAs that integrates on the same IC up to two PowerPC 405 processors with up to 200000 programmable logic cells.

Stephen Craven and Peter Athanas (2008) have discussed the creation of a high-level development environment for reconfigurable designs that leverage an existing high-level synthesis tool to enable the design, simulation, and implementation of dynamically reconfigurable hardware solely from a specification written in C. The authors have described a new approach to application development that leverages a commercial High Level synthesis (HLS) tool, integrated embedded processors and provided models of communication and reconfiguration.

Hardware/Software partitioning can improve the performance and in some cases even reduce power consumption. Hardware/Software partitioning techniques for SoC platforms composed by a microprocessor and FPGA have been developed recently. Sudarshan Banerjee et al (2006), have presented a heuristic approach of Hardware-Software partitioning that simultaneously partitions, schedules and does linear placement of tasks on Xilinx Virtex series of devices. These works in the context of high-level synthesis have been based on pipelined hardware synthesis of application's critical loops on coarse and fine grained reconfigurable coprocessors. The nature of these architectures is more of a software compilation problem than a hardware one, since applications are mapped to fixed architectures.

Mapping onto reconfigurable array architecture involves more constraints than high level synthesis design. The existing approaches are based on compilation approaches from the DSP world (e.g., dataflow graph of an unrolled loop being mapped onto the array, directly) or high level synthesis. However, these approaches are not able to completely exploit the parallelism and flexibility offered by the reconfigurable processor arrays. The efficient mapping of algorithms on reconfigurable architectures requires data dependency analysis. Porting existing, highly optimized C code to such an environment is a time consuming task and often ends in completely rewriting

the code from scratch. Furthermore, most existing tools do not allow high level program transformations in order to match the input program to given architecture constraints (like maximum available memory or I/O bandwidth), or only to a limited extent. Many existing algorithms try to avoid the restrictions of sequential languages by using different programming and execution models.

2.11 SUMMARY

A review of various design approaches in reconfigurable computing system for multistandard communication architectures and their pros and cons have been discussed in this chapter. Also, efficient implementation techniques of OFDM and WCDMA receiver architectures and mapping applications on reconfigurable array architectures have been investigated.