Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2016) 24: 1176 – 1193

doi:10.3906/elk-1307-129

An experimental study of coarse-grained reconfigurable system-on-chip-based software-defined radio

Janakiraman NITHIYANANTHAM*, Nirmal Kumar PALANISAMY
Department of Electronics and Communication Engineering, Faculty of Information and Communication Engineering, Anna University, Chennai, Tamil Nadu, India

Received: 16.07.2013 • **Accepted/Published Online:** 22.02.2014 • **Final Version:** 23.03.2016

Abstract: Software-defined radio (SDR) research deals with a mixture of hardware and software technologies, where RF operating parameters and components are to be set or altered by modifiable software or firmware. This paper describes the coarse-grained reconfigurable array (CGRA) implementations of SDR architecture. This architecture is an extension of traditional SDR in complex adaptation strategies, such as highly reliable communications and efficient utilization of the resources and spectrum upgrade, through its internal states (performance) and hardware architecture. The proposed CGRA-based SDR implementation is based on dynamic partial reconfiguration methodology, which has the capability of reusing the same hardware module to handle different algorithms. This CGRA-based SDR provides greater flexibility and adds new abilities without additional cost. Initially, the SDR system was simulated in the Agilent SystemVue environment to analyze the error boundaries of the proposed SDR architecture. Then the SDR system was coded in the Verilog hardware description language and implemented on top of CGRAs such as the MOLEN, MORPHOSYS, and ADRES reconfigurable system-on-chip (SoC) architectures. These SoC architectures were installed within the Xilinx Virtex 5 field-programmable gate array to analyze the performance of SDR architectures in terms of area utilization, operational speed, power optimization, reconfiguration time, coprocessor execution time, preemption support, and relocation support of the system. The performance analysis indicates that the ADRES SoC architecture is suitable for dynamic partial reconfiguration and the MOLEN SoC architecture is more suitable for power, area, and speed requirements and low circuit complexity compared to other architectures.

Key words: Software-defined radio, reconfigurable architecture, field-programmable gate array, system-on-chip, MOLEN, MORPHOSYS, ADRES

1. Introduction

Reconfigurable computing provides greater flexibility in dynamic reconfiguration of computation and communication resources such as spatial computing [1]. Coarse-grained reconfigurable arrays (CGRAs) are common examples of run-time reconfigurable systems, whereas field-programmable gate arrays (FPGAs) are examples of load-time configurable computing systems. RaPiD [2] and Matrix [3] are the earlier CGRAs, which are one-dimensional and two-dimensional structures of resources such as arithmetic and logical units (ALUs), multipliers, memories, and static routing networks. The FPGA fabric is an arrangement of a group of look-up tables (LUTs), block memories, and word-wide multipliers with tightly coupled horizontal and vertical interconnect topology [4]. A bitwise data format is used as a configuration file to specify the functionality of an

*Correspondence: janakiramanforu@yahoo.com

FPGA. CGRAs are used in the application of digital signal processing and digital communication techniques [5]. Hence, they are configured by multibit data words. This paper deals with an application of a modern digital communication system called software-defined radio (SDR).

SDR is a flexible architecture that is suitable for and applicable to many radio standards. It allows implementation of the signal processing algorithms in software instead of hardware [6]. Hence, the user-defined programs are handling different types of radio signals and communication protocols [7] without additional hardware circuits. The user has the freedom to set various communication properties, such as desired frequency, bandwidth, modulation, and data rate, simply by loading the suitable software [8]. Multimodulation (i.e. orthogonal frequency division multiplexing, OFDM) and multiantenna (i.e. multiinput-multioutput, MIMO) techniques are used to enhance the flexibility of SDR [9]. MIMO-OFDM has grown to be the most popular communication system in high-speed communications [10]. Here, the bandwidth is divided into many carriers (multicarrier), and each subcarrier is modulated by a low data-rate stream [11]. This multicarrier transmission technique is used in most modern wireless communication and cognitive radio systems [12].

This paper gives a detailed study of the implementation of the SDR architecture in popular reconfigurable heterogeneous CGRAs, namely MORPHOSYS (Morphoing system) [13], MOLEN (polymorphic processor) [14], and ADRES (architecture for dynamically reconfigurable embedded systems) [15]. Initially, the proposed SDR model is designed and simulated in the Agilent SystemVue software [16]. This model is then coded in Verilog HDL to implement on top of the CGRA-based system-on-chip (SoC) architecture. This project achieves greater functionality with a simpler hardware design. It is possible to reduce the cost of additional features by the dynamic partial reconfiguration (DPR) process when not all of the logic circuits are used at all times. The circuit and function of this proposed reconfigurable system can be customized at the application level and phase level over time. Hence, this project handles the trade-off between flexibility and performance in a proficient way.

This paper is organized as follows. Section 2 describes the proposed SDR system model. Section 3 presents the proposed MIMO set-up of this project. Section 4 gives the various modulation techniques used in this SDR system. Section 5 gives the simulation results of the proposed SDR model. Section 6 explains the proposed CGRA-based SoC implementation. Section 7 presents the reconfiguration process of this proposed CGRA design. Section 8 explains the experimental methodology of the project and analyzes the experimental results. Section 9 concludes the paper.

2. Proposed system description

The transmitter section of the SDR architecture is shown in Figure 1. First, the analog input signal is converted into fixed-point digital data (fixed-point constant) with 4-bit integer word length using sampling and quantization techniques. The precision value of these output data (output precision mode) is defined by the user. Second, the hardware-based digital modulator is used to map the amplitude, in-phase signal (I), and quadrature-phase signal (Q) in each subcarrier. Next, the frequency converter (Fr-Co) module is used to convert the various modulated signals, such as binary phase-shift keying (BPSK), quadrature phase-shift keying (QPSK), 16-quadrature amplitude modulation (QAM), and 64-QAM into the same frequency of 9.5 MHz (i.e. OFDM signal). Then a bandpass Chebyshev filter of 9.4 MHz with a ripple factor of 1 is used to maintain the conditions of orthogonality in the subcarriers. Finally, the signal mixer circuit is used to combine these data signals with carrier signals of 3.4 GHz and 7 dBm of power generated by a local oscillator unit. Here, the

conversion gain and noise figure are assumed as 1 and 0, respectively. Then the signal is transmitted through a 4 × 4 MIMO antenna set-up with additive white Gaussian noise (AWGN) and multipath Rayleigh fading channels.

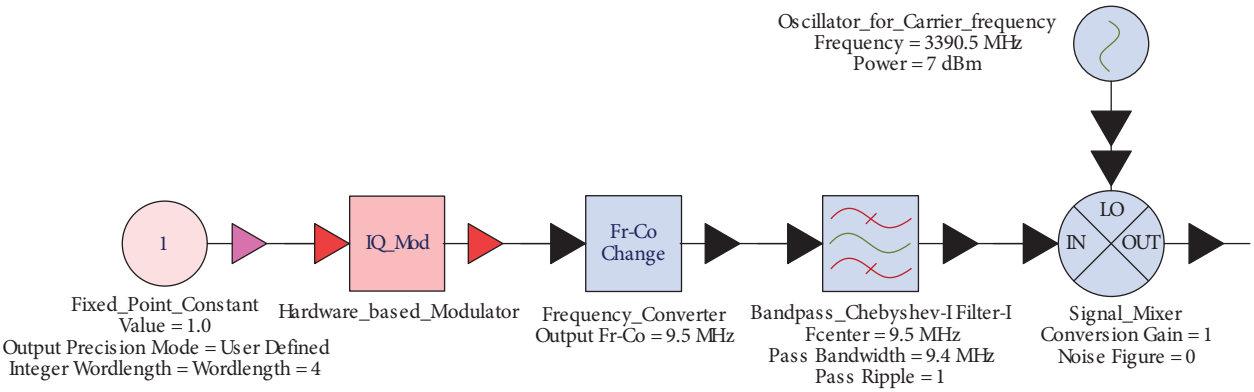

Figure 1. Transmitter section of SDR.

The receiver section (as shown in Figure 2) performs the reverse operation of the transmitter section. In the receiver section, first, a bandpass Chebyshev filter of 12 MHz with a ripple factor of 1 is used to receive the signals from 4 × 4 MIMO antenna set-ups. Second, a nonlinear preamplifier with a gain of 22.387 ($10^{27/20}$) is used to reduce the effects of noise and interference signals (i.e. unwanted spikes) and to maintain the process synchronization between transmitter and receiver sections through impedance matching. Again, a bandpass Chebyshev filter of 12 MHz with a ripple factor of 1 is used to retrieve the original data signals with the properties of OFDM. Then a nonlinear power amplifier with a gain of 3.981 ($10^{12/20}$) is used to amplify the data signals in terms of power. Finally, the received data signals are analyzed using an Agilent vector signal analyzer (VSA). The various parameters of the proposed SDR are given in Table 1.

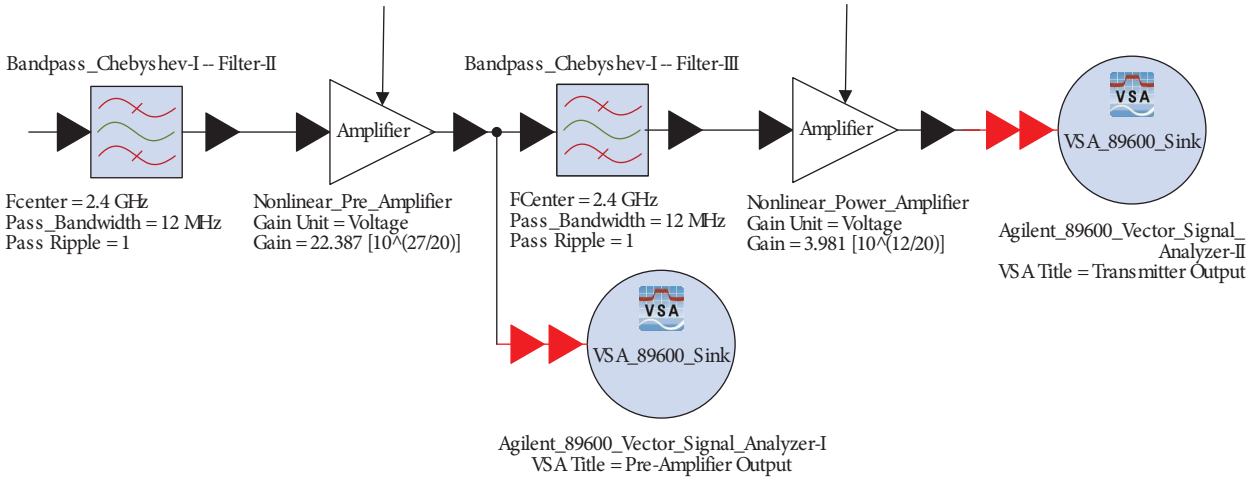

Figure 2. Receiver section of SDR.

Table 1. Design parameters of the SDR transceiver.

Configuration	Data
Operating frequency (F_S)	200 MHz
Modulation scheme	BPSK, QPSK, 16-QAM, 64-QAM
Coding type and coding rate	Convolution and 3/4 coding rate
Data rate	6, 12, 48, 54 Mbps
Number of subcarriers (N_{SC})	128
Number of data subcarriers (N_{DS})	108
Number of pilot subcarriers (N_{PS})	20
OFDM symbol period (T)	80 cycles (18.62 μs)
Cyclic prefix (T_C)	16 cycles (6.14 μs)
Bandwidth (B_W)	40.56 MHz
Subcarrier frequency (F_S)	234.48 KHz
Number of transmit antennas	4
Number of receive antennas	4
Maximum transmit power	1 W
AWGN PSD	–100 dB/Hz to 80 dB/Hz
BER	1e-3
Number of pipeline stages	24
Pipeline latency (μs)	0.39
Packet size	1000-byte packet length
Channel model	150-ns delay spread

3. Proposed MIMO setup

Wireless communication using the MIMO technique provides greater spectral efficiency by dividing an available total transmitted power into multiple spatial paths (or modes) and driving each mode with equal capacity. The introduction of additional spatial channels and space-time coding techniques is used to achieve this high spectral efficiency, with a much lower required energy per information bit [17]. In this proposed work, a 2-(bit/s)/Hz link is used with independent modulated (BPSK, QPSK, 16-QAM, and 64-QAM) sequences for each transmitted antenna with the help of a common local oscillator called "space diversity". Hence, a high rate signal is split into multiple lower rate streams and transmits those streams from different transmit antennas in the same frequency channel, using an Agilent vector signal generator (VSG) module.

The MIMO receiver gets multiple independent faded copies of the same information symbol. This space correlation property of the radio channel can be used to improve the reliability of the link. The diversity-combining technique is applied to combine these multiple received signals into a single improved signal. In this work, the selection combining technique is used to select the strongest signal in terms of signal-to-noise ratio (SNR) among all the received signals, as shown in Figure 3.

4. Modulation techniques for SDR

Modulation is a process by which a communication signal that contains information is combined with another signal called the carrier signal. There will be identical performance at identical power levels. This paper considers the basic modulation techniques used in the mobile and wireless systems. The modulation scheme for SDR is proposed based on the best reconstructed signal quality for each average SNR. The modulation technique is evaluated when the system is subject to noise and interference in the channel (Rayleigh multipath

fading channel). The processing core of SDR uses a sequential block-processing approach, in which each layer can be added or removed as required to create a flexible SDR architecture. In the receiver section, the distortion of waveforms caused by the channel and the effects of noise as well as interference in the received signal are eliminated by the bandpass Chebyshev filter. The finite bandwidth of the channel leads to distortion in the ideal signal. It also introduces noise from various sources, and the channel may also attenuate the input signal. Each of the effects mentioned above will affect the bit error rate (BER) and SNR of the received signal. The effects of attenuation and noise can be mathematically modeled by adding AWGN. The effects of finite bandwidth can be modeled with a simple filter.

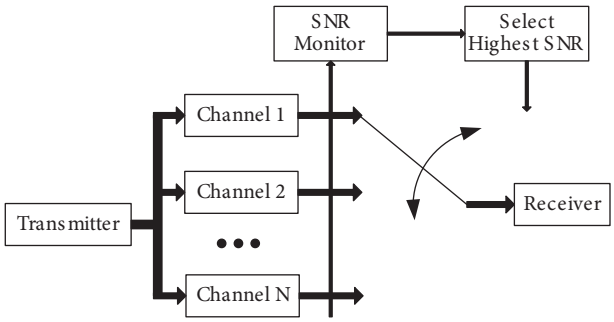

Figure 3. Diversity-combining technique in MIMO receiver.

In SoC implementation, the major problem is to reduce the implementation area of the decoding hardware. The QPSK technique supports only 4 phases: 0°, 90°, 180°, and 270°. Hence, the implementation of a 45° phase difference is harder between quarter components. This can be achieved by simply rotating the signal points of QAM by 45° clockwise. This rotation process does not affect the magnitude of the signal points and transmit power, but it provides a reasonable improvement in gain value. The QAM technique has the ability to implement a 45° phase shift in both the transmitter and receiver [18]. It has been widely used in adaptive modulation practices because of its efficiency in power and bandwidth. QAM is used to achieve a good combination of high data rates, efficient use of available bandwidth, low error rates, and rapid demodulation. In BPSK, the transmitted signal is a sinusoid of fixed amplitude. Hence, it has one fixed phase when the data is at one level, and an 180° phase shift at another level of data.

5. Simulation results

A simulation is a replication of the operation of a real-world process over a time duration. Initially, the simulation process requires a model that represents the key characteristics, functions, and behaviors of a particular physical component or system. In this work, the SDR system model is represented in a combination of circuit description and algorithmic flow format. The behavior of a proposed SDR system is analyzed by changing variables in the simulation process. It is a method for performance optimization to improve the gain of the proposed SDR system under the eventual real effects of alternative conditions and flow of action. In each simulation process of the proposed SDR system, the following behaviors are analyzed in detail:

1) Spectrum of the received OFDM signal (Figure 4)

   This shows the spectra of subcarriers, which are represented by a sequence of Sinc functions with alternating polarity or zero crossing points in identical spaces. Then the high-frequency band usage is achieved using the technique of spectrum overlapping with null inter-subcarrier interference. Hence, the total power spectrum shape is close to square.

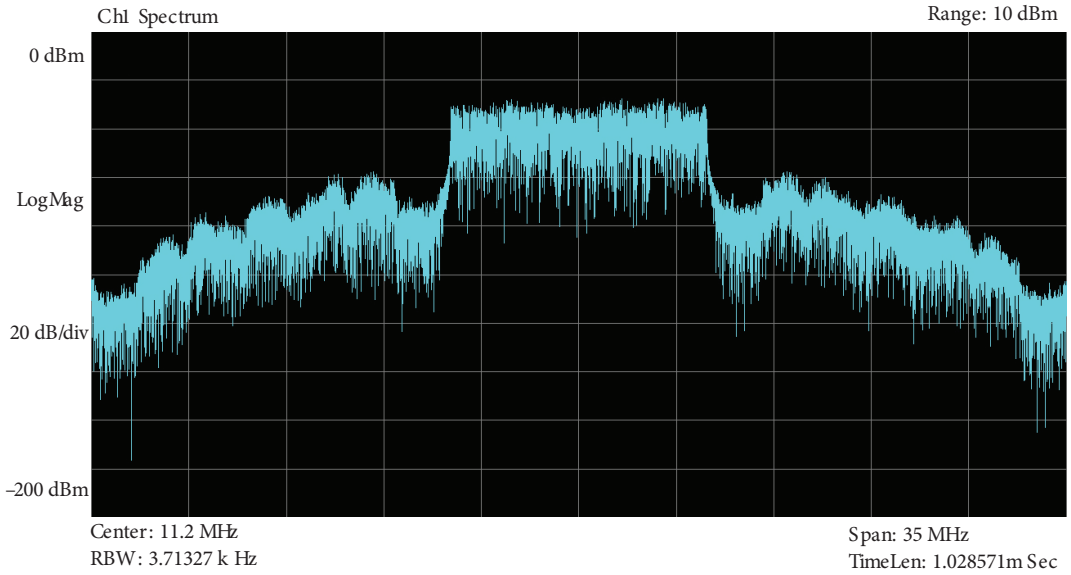

Figure 4. OFDM spectrum.

2) Frequency response of the received OFDM channels (Figure 5)

In the frequency domain, the spectrum of subcarriers allows the signal to go through without bending or distortion, and it does not allow any information in particular bands, which are deep fades frequencies. This form of channel frequency response is called frequency selective fading.

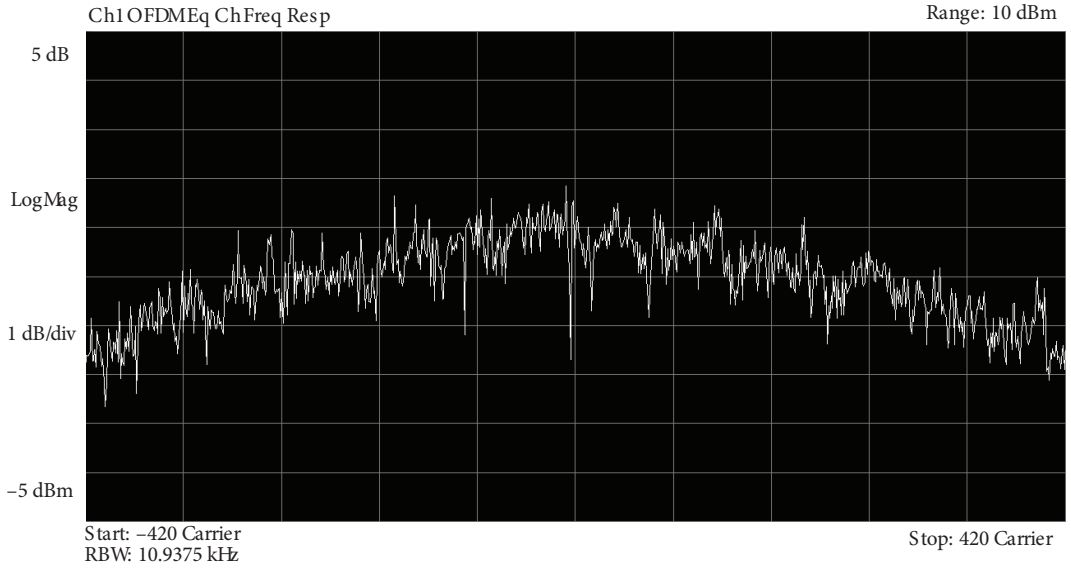

Figure 5. OFDM-equivalent channel frequency response.

3) Error vector spectrum of the received OFDM signals (Figure 6)

The delay mismatch error (i.e. electrical cable length or traces and timing skew) occurs when the 'I' signal differs from the 'Q' signal. This delay linearly increases the center subcarrier and the phase noise,

which affects the constellation diagram. This phase signal error may occur as a function of the subcarrier number or frequency, known as an error vector spectrum. This plot shows the function of the carrier, which represents both the RMS and the individual errors in every symbol. Hence, the outer carrier symbols have more phase error than the inner carrier symbols.

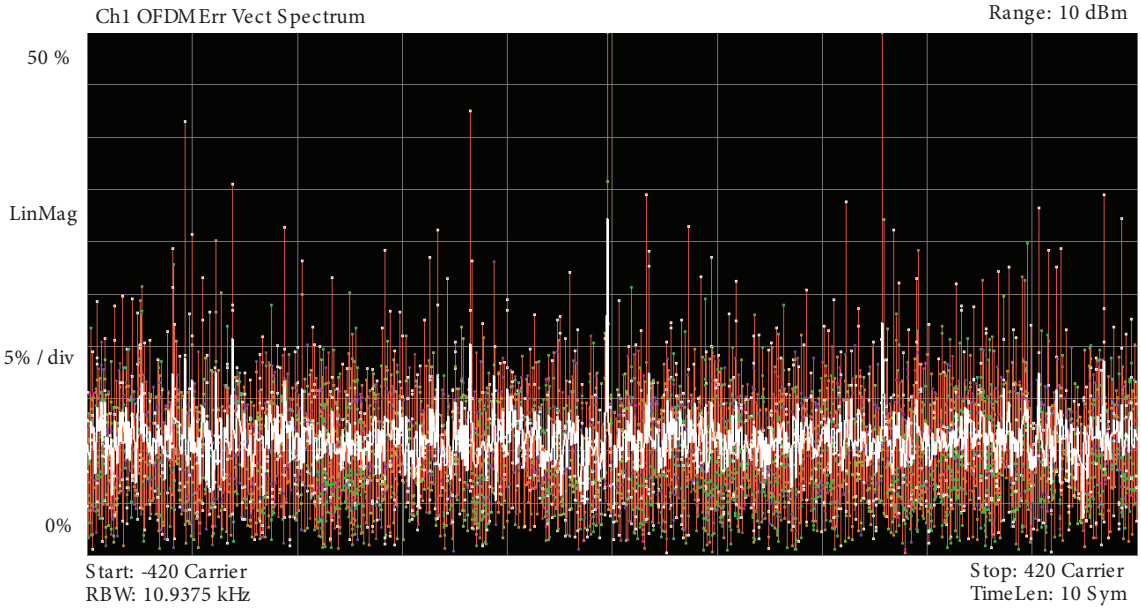

Figure 6. OFDM error vector spectrum.

4) Distribution of the received OFDM signals in quadrature phases (Figure 7)

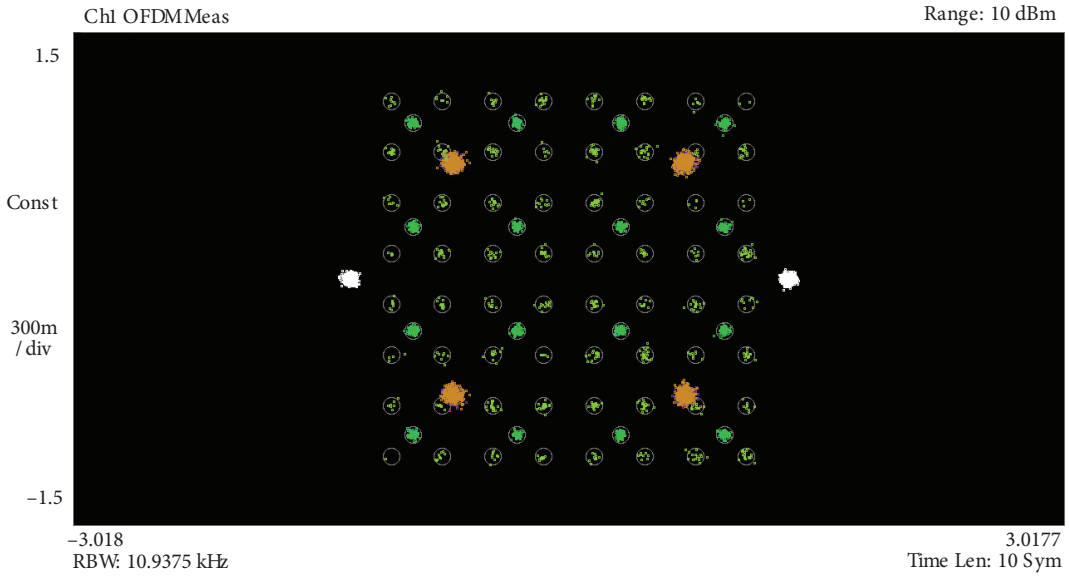

Figure 7. OFDM meas trace data.

The constellation of received OFDM signals is disturbed due to the addition of a delay signal to an original signal. An equalizer is used to correct these disturbances. Here the frequency error is expressed

as a cumulative phase error that linearly increases or decreases with time. It has been visualized as a spinning constellation diagram.

5) SNR vs. average BER (Figure 8)

This shows that the average BER is 10^{-6} for 13, 18, 21, and 23 dB of SNR of BPSK, QPSK, 16-QAM, and 64-QAM, respectively. It also indicates that the received signal has good signal strength and that its error rate is low.

6) SNR vs. packet error rate (Figure 9)

This shows that the average packet error probability varies between 0.1 and 0.7 for the average SNR of 15 dB. It also indicates that the received signal has low packet error rate (PER) and good signal strength.

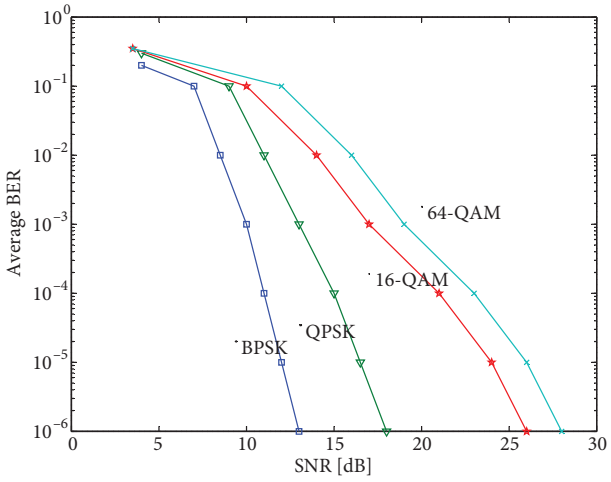

Figure 8. SNR vs. average BER.

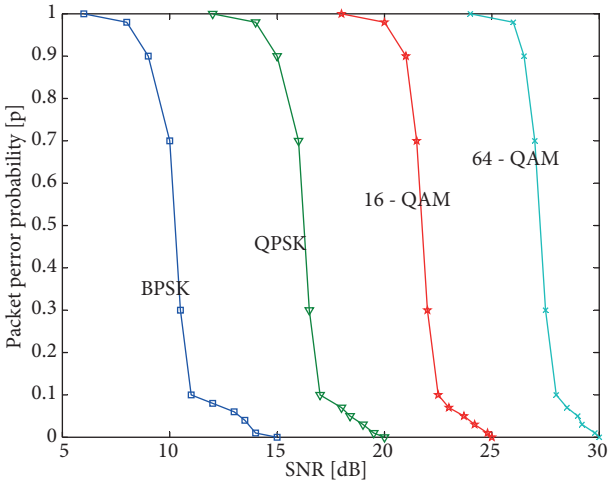

Figure 9. SNR vs. packet error probability.

6. SoC implementation

Heterogeneous CGRAs adapt the smart computing technique, which combines the flexibility of software with standard hardware modules and is capable of high-speed data processing. FPGAs are a suitable hardware platform for the implementation of heterogeneous CGRA compared to microprocessors, microcontrollers, and custom hardware (i.e. application-specific integrated circuits). These CGRAs can adapt the required hardware architecture (i.e. custom logic) during run-time by downloading a new circuit on the configurable logic block. The traditional reconfigurable computer architecture uses different methods of configuration, namely HDL, electronic design automation, electronic system level, 'C'-based language, and graphical tools such as star bridges [19,20]. In this paper, the Verilog HDL language model is used to configure the CGRA-based SoC architecture through Xilinx software. Here, 3 popular SoC architectures, MOLEN, MORPHOSYS, and ADRES, are considered for real-time applications on the FPGA.

At the initial stage of the SoC implementation, the SDR architecture is partitioned into multiple modules. This partitioning process is considered as a hypergraph treatment of the transition probability matrix-based Markov chain process [21]. Next, this SDR architecture module is automatically mapped and scheduled on a CGRA-based SoC structure using the hardware/software codesign and coverification techniques [22].

6.1. MOLEN polymorphic processor

The MOLEN reconfigurable processor (Figure 10) consists of 2 major components. These are the core processor, which is a general purpose processor, and the reconfigurable processor, which is used for special-purpose reconfigurable applications. The data exchanges between these 2 processor units are handled by the exchange registers. The register file unit is used as a temporary storage unit for the core processor. The user instructions are given from the main memory unit to the arbiter unit through the instruction fetch unit. The user data are fetched from the main memory unit to the data memory MUX/DEMUX unit through the data load/store unit. The reconfigurable processor is further subdivided into the reconfigurable microcode unit, which is used for handling microcode instructions, and the custom configuring unit, which consists of reconfigurable hardware, such as the FPGA. Microcode instructions are hardware-level instructions involved in the execution of high-level machine code instructions in the processing units of many internal logic circuits. Microcodes may be feasible to reduce the complexity of the electronic circuitry using a set of multistep instructions.

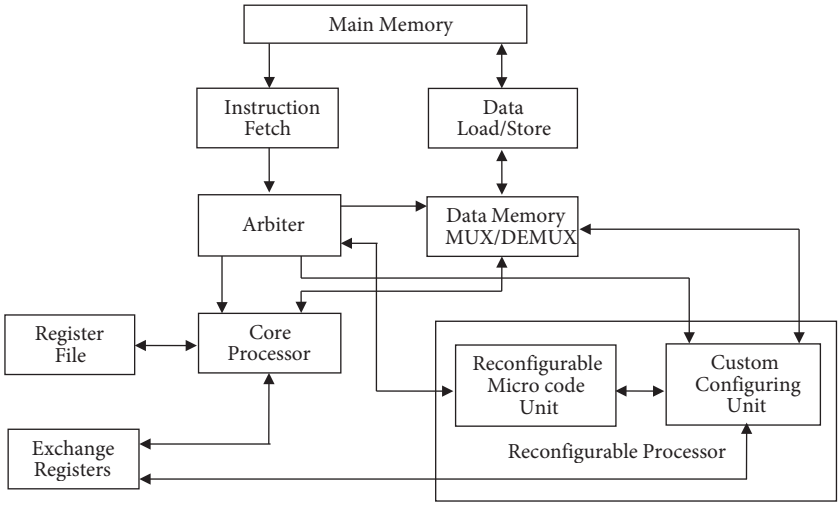

Figure 10. MOLEN reconfigurable processor.

The MOLEN architecture utilizes both microcode and custom-configuring hardware for high-speed applications. It considers all kinds of processor requirements, from embedded systems to supercomputers. The execution of reconfigurable hardware (ranging from a single instruction to a piece of application code) is divided into 2 logical phases. The reconfigurable hardware is configured in the first phase, and the fixed (or) core units are executed in the second phase. The microcode instructions are utilized to perform both the reconfiguration process and the execution of the core units. Here, the frequently utilized microcode resides permanently within the fixed part of an on-chip storage facility, and the nonfrequent microcode is paged into the pageable part of the same storage unit. Since this approach is generic, various applications can utilize the proposed processing capabilities. The wireless transceiver model (i.e. SDR) is implemented on top of the MOLEN reconfigurable processor using automatic mapping and scheduling processes.

6.2. Morphing system

The MORPHOSYS system is an array of tightly coupled reconfigurable cells and is closely associated with the core processor known as the tiny reduced instruction set computing (TinyRISC) processor, which is a million instructions per second (MIPS) processor (Figure 11). The reconfigurable cell (RC) array is composed of four

4 × 4 cell quadrants. The TinyRISC processor monitors and controls both the general-purpose operations and the reconfigurable cells. Each reconfigurable cell contains four 16-bit registers, a 32-bit context register, a shift register, input/output (I/O) multiplexers, and an ALU, as shown in Figure 12. The frame buffer is an internal data memory that contains 2 sets of logical groups, set 0 and set 1. The direct memory access (DMA) controller initiates the loading of configuration bits from the main memory to the context memory and the data transfers between the main memory and the frame buffer. The data operands are sent from the frame buffer to the RC array by a 128-bit operand bus.

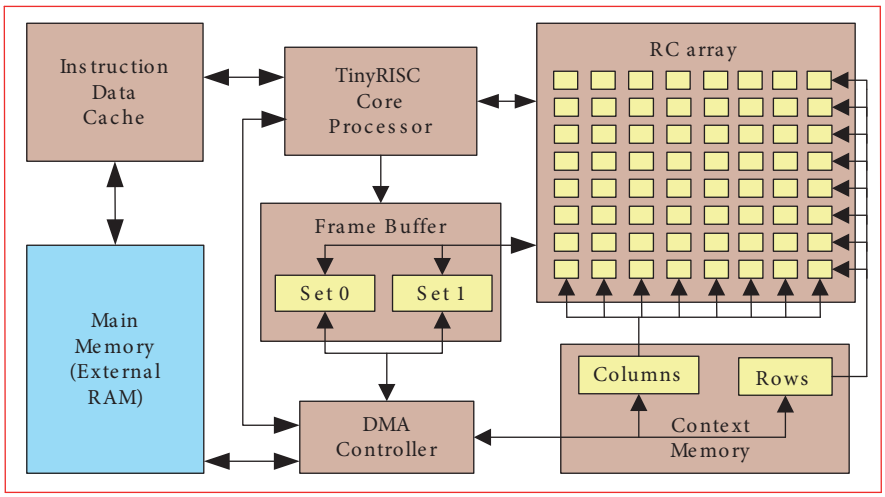

Figure 11. MORPHOSYS reconfigurable processor.

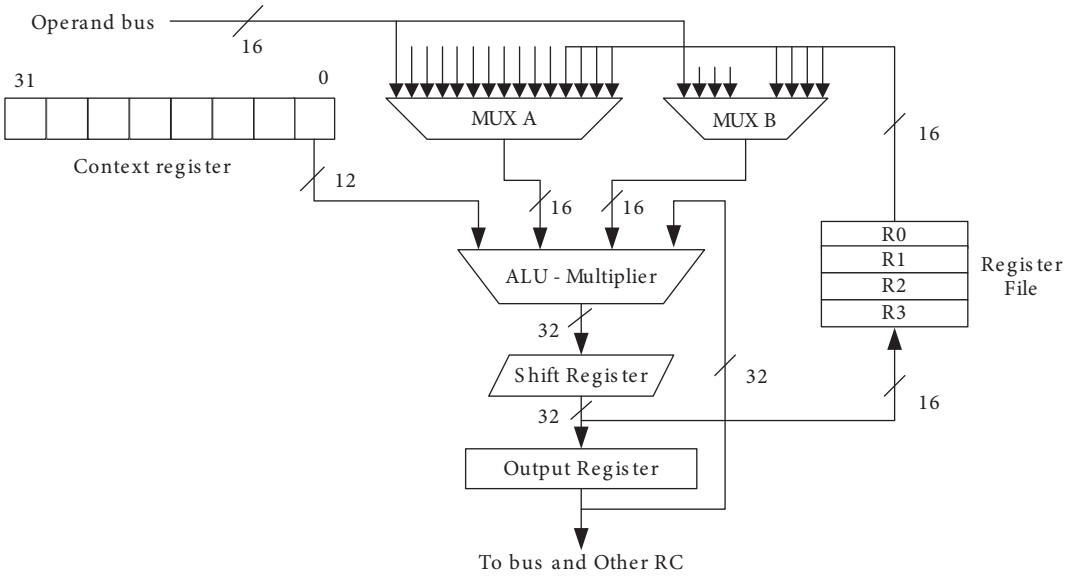

Figure 12. Architecture of the reconfigurable cell (RC).

MORPHOSYS is a reconfigurable single-instruction multiple-data (SIMD) architecture mainly composed of the RC array, TinyRISC processor, frame buffer, context memory, and DMA module. Hence, the dynamic reconfiguration can be achieved by context updating of reconfigurable cells to implement the SDR architecture on top of the MORPHOSYS reconfigurable processor, using the automatic mapping and scheduling process.

6.3. ADRES system

The ADRES system is a suitable platform for dynamic and partial reconfigurable operations, due to its flexibility and a high degree of design freedom. It is an architecture design template for dynamically reconfigurable and statically scheduled CGRAs, as shown in Figure 13. It consists of a very long instruction word (VLIW) processor and a group of reconfigurable resources known as the coarse-grained array (CGA), with register files (RFs), functional units (FUs), and interconnects. The VLIW processor consists of a control unit (CU), which is responsible for the fetch and dispatch of instructions, and which controls the switching of operating modes between the VLIW and the CGA units. It also consists of an instruction cache memory (ICache), global program RFs (PRF), and global data RFs (DRF) to speed up the execution process. The VLIW processor can be executed by a single sequential logic thread for arithmetic operations, logic operations, load/store operations, and predicate computing instructions.

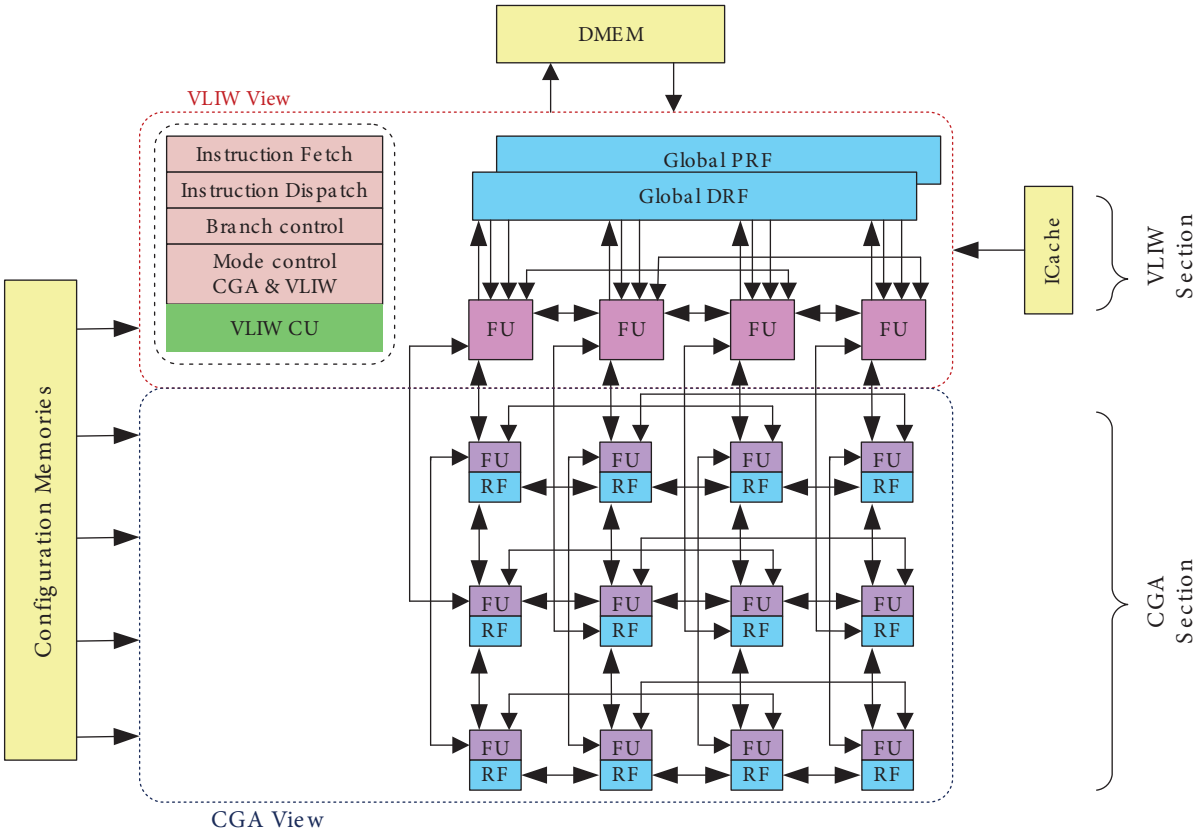

Figure 13. ADRES SoC platform.

The CGA architecture uses the modulo-scheduling approach to run a large number of vector functional units in simultaneous placement, routing, and scheduling subproblems in the target application. This reconfigurable array is used to accelerate the dataflow of application kernels in a highly parallel way, while the VLIW processor executes the other portions by exploiting instruction-level parallelism (ILP). The multiplexers, buses, and point-to-point connections are used to interconnect the FU and RF of the CGA structure. The external memories, such as configuration memories and dynamic memory (DMEM), are used to accelerate the dynamic reconfiguration process.

Based on user demand applications such as SDR, the ADRES platform is capable of adopting high data memory bandwidth. The entire SDR application is automatically mapped and scheduled by the VLIW processor using the traditional ILP compilation techniques. The communication and synchronization task between the VLIW processor and the CGA architecture is controlled and monitored by a VLIW processor using some special instructions.

7. Reconfiguration process

The process of altering the structure or function of a device at run-time is called dynamic reconfiguration. The hardware design or circuit may change in response to the demands placed upon the system at deployment time, during execution, or between execution phases. Normally, a bit stream is used for deployment of a device or circuit into the reconfigurable system at run-time. The dynamically reconfigurable system supports changes in the hardware during run-time, since it has flexibility similar to that of software [23]. It may lead to better performance and smaller system size. The coarse-grained architecture requires less configuration time and lower potential energy compared to fine-grained architecture due to the lesser number of elements, enough to be programmed or addressed.

DPR is a technique that allows one part of the device to be reconfigured without disturbing the active computation of other parts. Hence, this DPR evaluation reduces the overall area constraints of a circuit by removing potentially irrelevant hardware within the implementation [24]. At the initial stage of the DPR process, the partial bit streams are created based on design constraints and stored in the configuration memory. In the future, this bit stream can be compressed to ensure lower power and energy consumption.

This project uses the reconfigurable array as a processing accelerator. Here, the different configurations, including various components of the MIMO-OFDM transceiver architecture, can be programmed or executed in different phases. Hence, customization or optimization of the hardware is possible at the application level and phase level over time.

8. Experimental methodology and analysis

At the initial stage, both the MOLEN reconfigurable processor and the SDR architecture are considered as 2 different partitions of the SoC. The installation preferences of these partitions are given to the partition assignment unit. Then each partition is synthesized separately by Xilinx ISE 12.2, and the synthesized results are sent to the partition merge unit. First, the MOLEN reconfigurable processor is installed as a portion of SoC using a partial bit-stream, and then the SDR architecture is installed within the MOLEN reconfigurable processor using another partial bit-stream. Here, the reconfiguration observer unit detects situations where reconfigurations need to be performed, and then it sends that information to the configuration scheduler module through the settings and the assignment unit, as shown in Figure 14. All the synthesized partitions are merged as a single bit-stream and that is sent to the fitter unit, which is responsible for checking user constraints, including the floorplan, place, and route of the modules. Then the simulation process (timing analyzer) and synthesis process (assembler) are carried out to verify the functionality of the SoC as per user-defined constraints. Finally, the Xilinx Virtex-5 FPGA (xc5vlx110t-3ff1136) kit is used as a configuring platform for this dynamic and partial reconfigurable SoC implementation through the joint test action group (JTAG) configuration port and the built-in 3.3 V I/O voltage rail. The final results are analyzed in terms of area, power, speed, reconfiguration time, coprocessor execution time, preemption support, and relocation support [25]. Similarly, the MORPHOSYS and ADRES reconfigurable processors are also analyzed, and the final implementation results are summarized in Tables 2 and 3.

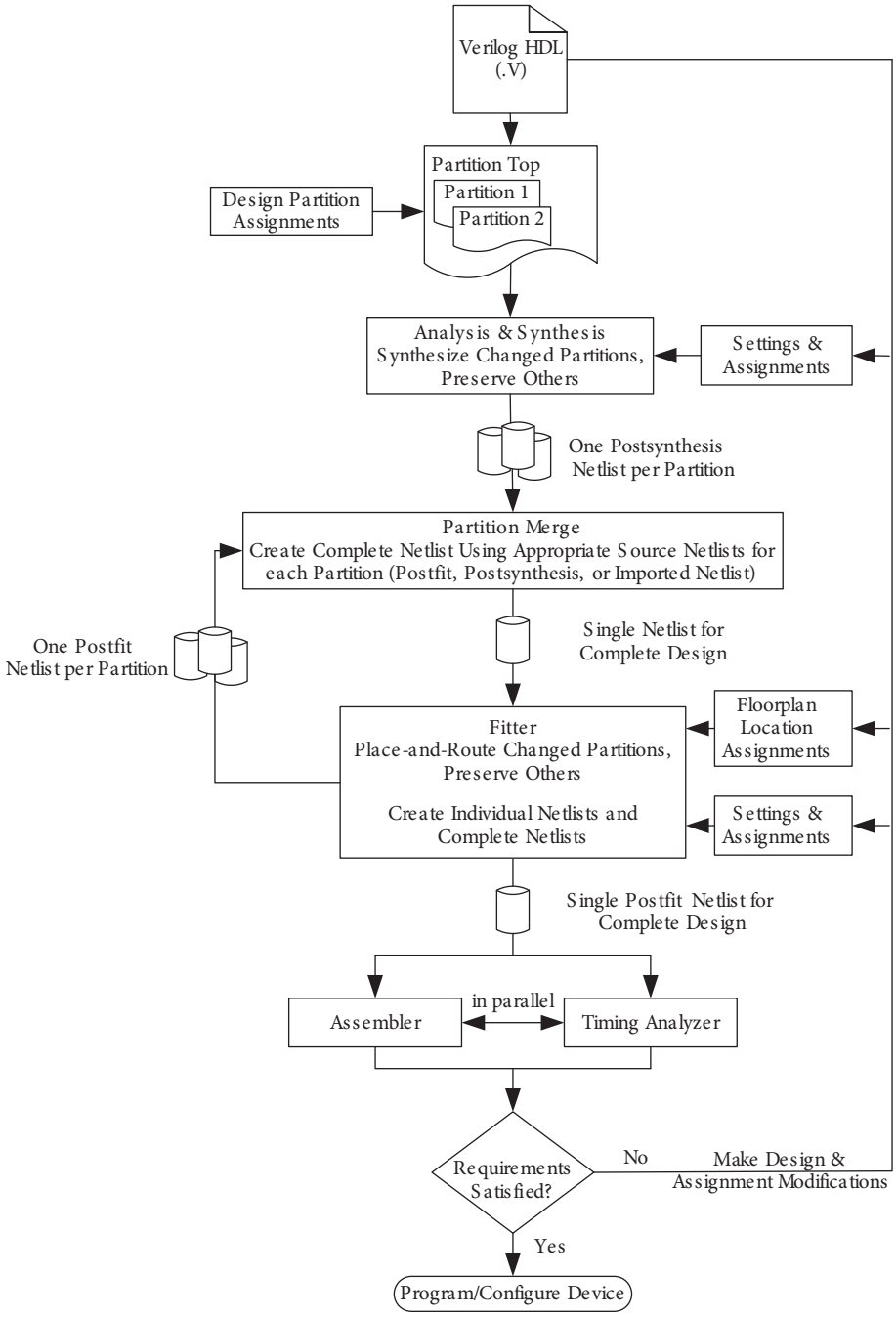

Figure 14. The flow of dynamic and partial reconfiguration.

8.1. SoC implementation analysis

Normally, in SoC implementation, there may be a trade-off between area, speed, and power requirements. This may be solved by the user or designer based on their applications. Here, the SDR architecture is taken as a testing application and mapped to the CGRA architectures (Table 2).

The total requirement of implementation area for the MOLEN SoC architecture was 42.4%, whereas ADRES SoC architecture and MORPHOSYS SoC architecture required 43.7% and 55.8%, respectively. The

Table 2. SoC implementation.

Area analysis				Speed analysis			Power analysis			
Resource type	Available (No.)	Resource utilization		Maximum frequency (MHz)	Minimum execution period (ns)	Minimum input arrival time before clock (ns)	Maximum output required time after clock (ns)	Quiescent power (W)	Dynamic power (W)	Total power (W)
		(No.)	%							
MOLEN										
Number of slice registers	69120	2080	3.0	260.76	3.835	3.223	4.301	1.22283	0.0280	1.22563
Number of slice LUTs	69120	2848	4.1							
Number of fully used LUT–FF pairs	9	3	33.3							
Number of bonded IOBs	640	13	2.0							
ADRES										
Number of slice registers	69120	2432	3.5	234.52	4.264	3.718	4.935	1.22984	0.0287	1.25854
Number of slice LUTs	69120	3198	4.6							
Number of fully used LUT–FF pairs	9	3	33.3							
Number of bonded IOBs	640	15	2.3							
MORPHOSYS										
Number of slice registers	69120	2698	3.9	212.04	4.716	4.372	5.404	1.25061	0.0292	1.27981
Number of slice LUTs	69120	3315	4.8							
Number of fully used LUT-FF pairs	9	4	44.4							
Number of bonded IOBs	640	17	2.7							

requirement of total execution time (T) was also less in the MOLEN SoC (3.835 ns) compared to the ADRES SoC (4.264 ns) and MORPHOSYS SoC (4.716 ns) architectures. The execution period was inversely proportional to the execution speed (f) or frequency $(f = 1/T)$ of the VLSI architecture. Hence, the maximum supportive frequency or speed was also high in the MOLEN SoC (260.76 MHz) compared to the ADRES SoC (234.52 MHz) and MORPHOSYS SoC (212.04 MHz) architectures. The execution speed was directly proportional to the total power consumption (P) of the VLSI architecture. The total power consumption includes the static power (P_s) and dynamic power (P_d) of the architecture (i.e. $P = P_s + (P_d = \alpha C V_{DD}^2 f)$). Here, the total requirement of

power for the MOLEN SoC was 1.22563 W, whereas ADRES SoC and MORPHOSYS SoC required 1.25854 and 1.27981 W, respectively.

These CGRA-based SoC implementation results show that the MOLEN SoC architecture has better performance than the MORPHOSYS and ADRES SoC architectures in terms of high speed, low power, and less requirement of circuit area.

8.2. DPR implementation analysis

The entire SoC architecture with SDR implementation was analyzed under the condition of DPR (Table 3). Since this design consists of a higher number of advanced features that are never employed concurrently, all these elements need not be implemented at the same time. Hence, this technique is used to maximize the resource utilization percentile in SoC devices by configuring the features when it is needed for the application. This type of implementation needs prior knowledge about the reconfiguration level and reconfiguration time. The DPR implementation strongly depends on the design partitions of the architecture and the ability of the designer. Under the conditions of DPR, the MOLEN SoC architecture required 430 ms for total system reconfiguration and 15 ms for coprocessor execution, whereas ADRES SoC architecture required 365 ms and 12 ms, respectively, and MORPHOSYS SoC architecture required 460 ms and 17 ms, respectively. The proposed CGRA-based SDR architecture (MOLEN, ADRES, and MORPHOSYS) supports the preemption feature, which can temporarily interrupt a task from being carried out by a reconfigurable system, with the intention of resuming that task at a later time without requiring its cooperation during the DPR process. This proposed work (ADRES and MORPHOSYS) supports the relocation or rescheduling process for various components of a system to provide high performance in terms of power and area utilization.

Table 3. DPR implementation.

Parameters	MOLEN	ADRES	MORPHOSYS
Reconfiguration time (ms)	430	365	460
Coprocessor execution time (ms)	15	12	17
Preemption support	Yes	Yes	Yes
Relocation support	No	Yes	Yes

8.3. Experimental results

The results of the proposed work are listed below.

1. This work achieves greater functionality with a simpler hardware design [26]. It is possible to reduce the cost of additional features by DPR process when not all of the logic is used at all times (Sections 7 and 8.2).

2. This project uses efficient software management of the reconfigurable hardware against the traditional embedded processor or microcontroller-based system [27] (Section 8.1).

3. The trade-off between the maximization of the resource utilization percentile with higher operational speed and the minimization of the dynamic power consumption is handled in a proficient way [28] (Sections 7 and 8.2).

4. The optimization of hardware architecture is greatly achieved by the DPR process of pipelining and by parallel processing techniques at the runtime of the system [29] (Sections 7, 8.1, and 8.2).

5. The circuit and function of the reconfigurable system can be customized at application level and phase level over time (Sections 7 and 8.2).

9. Conclusion

This proposed SoC-based SDR architecture supports dynamic and partial reconfigurable embedded systems. It can utilize the same software and hardware modules for different logics and algorithms. This paper deals with BPSK-based, QPSK-based, 16-QAM-based, and 64-QAM-based SDR wireless communication systems. The DPR implementation results (Section 8.2) show that this dynamic and partial reconfigurable SoC-based SDR provides greater flexibility to add new features or modules, without the additional cost of software and hardware using preemption and relocation techniques. This proposed system has been successfully simulated in the Agilent SystemVue environment and reproduced in the MOLEN, MORPHOSYS, and ADRES reconfigurable SoC models on the Xilinx Virtex 5 FPGA-based development board. Its efficiency and performance were verified at the software as well as the hardware level. As per the power, area, and speed requirements and low circuit complexity (Section 8.1), the MOLEN SoC architecture is the best option for SDR realization compared to MORPHOSYS and ADRES SoC architectures. In terms of dynamic and partial reconfiguration property (Section 8.2), ADRES SoC architecture is the best choice for SDR realization. Therefore, the selection of a suitable SoC architecture is a trade-off between system performance and user flexibility.

Acknowledgments

This work was supported in part by the All India Council for Technical Education – Quality Improvement Programme Scheme 2010. Research and computing facilities were provided by Anna University and the K.L.N. College of Engineering.

References

[1] Ackley DH, Williams LR. Homeostatic architectures for robust spatial computing. In: IEEE 2011 Conference on Self-Adaptive and Self-Organizing Systems Workshops; 3–7 October 2011; Ann Arbor, MI, USA. New York, NY, USA: IEEE. pp. 91–96.

[2] Ebeling C, Cronquist DC, Franklin P. RaPiD—Reconfigurable pipelined datapath. In: Hartenstein RW, Glesner M, editors. International Workshop on Field-Programmable Logic and Applications. Berlin, Germany: Springer-Verlag, 1996. pp. 126–135.

[3] Mirsky E, DeHon A. MATRIX—A reconfigurable computing architecture with configurable instruction distribution and deployable resources. In: IEEE Symposium on FPGAs for Custom Computing Machines; 17–19 April 1996; Napa Valley, CA, USA. New York, NY, USA: IEEE. pp. 157–166.

[4] Minev PB, Kukenska VS. The Virtex-5 routing and logic architecture. In: Proceedings of the 18th International Scientific and Applied Science Conference of Electronics; 14–17 September 2009; Sozopol, Bulgaria. pp. 107–110.

[5] Lenart T. Design of reconfigurable hardware architectures for real-time applications–modeling and implementation. PhD, Lund University, Lund, Sweden, 2008.

[6] Mueck M, Piipponen A, Kalliojärvi K, Dimitrakopoulos G, Tsagkaris K, Demestichas P, Casadevall F, Péerez-Romero J, Sallent O, Baldini G et al. ETSI reconfigurable radio systems-status and future directions on software defined radio and cognitive radio standards. IEEE Commun Mag 2010; 48: 78–86.

[7] Ulversøy T. Software defined radio: challenges and opportunities. IEEE Commun Surveys Tuts 2010; 12: 531–550.

[8] Minden GJ, Evans JB, Searl L, DePardo D, Petty VR, Rajbanshi R, Newman T, Chen Q, Weidling F, Guffey J et al. KUAR: a flexible software-defined radio development platform. In: IEEE 2007 International Symposium on Dynamic Spectrum Access Networks; 17–20 April 2007; Dublin, Ireland. New York, NY, USA: IEEE. pp. 428–439.

[9] Raja J, Kannan M. VLSI implementation of high throughput MIMO-OFDM transceiver for 4th generation systems. Indian J Eng Mater Sci 2012; 19: 307–319.

[10] Yoshizawa S, Miyanaga Y. VLSI Implementation of a 600-Mbps MIMO-OFDM wireless communication system. In: IEEE Asia Pacific Conference on Circuits and Systems; 4–7 December 2006; Singapore, Singapore. New York, NY, USA: IEEE. pp. 93–96.

[11] Ogasawara Y, Odagiri S, Yoshizawa S, Miyanaga Y. Performance evaluation of environment-adaptive agent system in OFDM cognitive radio. In: International Symposium on Intelligent Signal Processing and Communication Systems; 8–11 February 2009; Bangkok, Thailand. New York, NY, USA: IEEE. pp. 1–4.

[12] Taha HJ, Salleh MFM. Multi-carrier transmission techniques for wireless communication systems: a survey. WSEAS T Commun 2009; 8: 457–472.

[13] Singh H, Lee MH, Lu G, Kurdahi FJ, Bagherzadeh N, Chaves Filho EM. MorphoSys: an integrated reconfigurable system for data-parallel and computation-intensive applications. IEEE T Comput 2000; 49: 465–481.

[14] Vassiliadis S, Wong S, Gaydadjiev G, Bertels K, Kuzmanov G, Panainte EM. The MOLEN polymorphic processor. IEEE T Comput 2004; 53: 1363–1375.

[15] Wu K, Kanstein A, Madsen J, Berekovic M. MT-ADRES: multithreading on coarse-grained reconfigurable architecture. Int J Electron 2008; 95: 761–776.

[16] Schiff M. Signal and algorithm development environment for SDR. In: IEEE Military Communications Conference; 28–31 October 2001; Tysons Corner, VA, USA. New York, NY, USA: IEEE. pp. 225–229.

[17] Zheng L, Tse DNC. Diversity and multiplexing: a fundamental tradeoff in multiple-antenna channels. IEEE T Inform Theory 2003; 49: 1073–1096.

[18] Häring L, Chen Y, Czylwik A. Automatic modulation classification methods for wireless OFDM systems in TDD mode. IEEE T Commun 2010; 58: 2480–2485.

[19] Jóźwiak L, Nedjah N, Figueroa M. Modern development methods and tools for embedded reconfigurable systems: a survey. Integration 2010; 43: 1–33.

[20] Marwedel P. Embedded System Design: Embedded Systems Foundations of Cyber-Physical Systems. 2nd ed. Dordrecht, Germany: Springer, 2011.

[21] Janakiraman N, Nirmal Kumar P. Multi-objective module partitioning design for dynamic and partial reconfigurable system-on-chip using genetic algorithm. J Syst Architect 2014; 60: 119–139.

[22] Lee J, Chung MK, Cho YG, Ryu S, Ahn JH, Choi K. Mapping and scheduling of tasks and communications on many-core SoC under local memory constraint. IEEE T Comput Aid D 2013; 32: 1748–1761.

[23] Koch D, Beckhoff C, Torrison J. Fine-grained partial runtime reconfiguration on Virtex-5 FPGAs. In: IEEE 2010 Annual International Symposium on Field-Programmable Custom Computing Machines; 2–4 May 2010; Charlotte, NC, USA. New York, NY, USA: IEEE. pp. 69–72.

[24] Di Carlo S, Gambardella G, Indaco M, Prinetto P, Rolfo D, Trotta P. Dependable dynamic partial reconfiguration with minimal area and time overheads on Xilinx FPGAS. In: Proceedings of the 23rd International Conference on Field Programmable Logic and Applications; 2–4 September 2013; Porto, Portugal. New York, NY, USA: IEEE. pp. 1–4.

[25] Rauwerda GK. Multi-standard adaptive wireless communication receivers: adaptive applications mapped on heterogeneous dynamically reconfigurable hardware. PhD, University of Twente, Enschede, the Netherlands, 2008.

[26] Roberto A. Design and implementation of software defined radios on a homogeneous multi-processor architecture. DSc, Tampere University of Technology, Tampere, Finland, 2013.

[27] Wu C, Cen F, Cai H. A high-performance heterogeneous embedded signal processing system based on serial RapidIO interconnection. In: IEEE 2010 International Conference on Computer Science and Information Technology; 9–11 July 2010; Chengdu, China. New York, NY, USA: IEEE. pp. 611–614.

[28] Krill B, Ahmad A, Amira A, Rabah H. An efficient FPGA-based dynamic partial reconfiguration design flow and environment for image and signal processing IP cores. Signal Process-Image 2010; 25: 377–387.

[29] Venkatasubramanian V. Hardware support for dynamic partial reconfiguration—accelerating multiple functions. MSc, Delft University of Technology, Delft, the Netherlands, 2011.