Software Defined Radio Principles and Platforms

A.M.Lalge[1] , M.S.Karpe[2] , S.U.Bhandari[3]

Abstract

Software Defined Radio (SDR) is defined as radio in which some or all of the physical layer functions are software defined. It is a radio system whose components are realized in software rather than in hardware. SDR system provides flexibility and portability. The main idea behind SDR is that more of the waveform processing can be implemented in reprogrammable digital hardware so a single platform can be used for multiple waveforms. Typical implementations of s SDR modems include a general purpose processor (GPP), digital signal processor (DSP) and field programmable gate array (FPGA). In SDR FPGAs can be used as both an interconnect layer and a general-purpose computational fabric implementing hardware acceleration units. The FPGA fabric can be used to offload either the GPP or DSP (or both). This paper discusses the principle of SDR and the platforms used for SDR implementation.

Keywords

SDR, DDC, DUC, FPGA, Hardware Acceleration

1. Introduction

SDR is universally accepted now. It was first proposed by Joseph Mitola in 1991. The SDR concept has been around for some time, but with recent advancements in electronics, SDR systems are becoming increasingly feasible. The focus now is shifting towards building maximally flexible and efficient systems. SDR is a collection of hardware and software technologies that enable re-configurable system architectures for wireless networks and user terminals. SDR provides an efficient and comparatively inexpensive solution to the problem of building multimode, multi-band, multifunctional wireless devices that can be adapted, updated or enhanced by using software upgrades. To achieve this purpose, the primary goal of SDR is to replace as many analog components and hardwired digital VLSI devices of the transmitter-receiver as possible with programmable devices. This includes: air interface, modulation and coding schemes and Data converters (ADC/DAC) [5]. The ultimate goal in radio transceiver design is to implement all transceiver functions in software. This topic presents principles of SDR and implementation of SDR on various platforms.

A.M.Lalge, PCCOE, Pune University
M.S.Karpe, PCCOE, Pune University
S.U.Bhandari, PCCOE, Pune University

A. Background and Relevance

The concept of SDR has been around for many years. SDR systems development has been driven by the evolution of radio communication systems from primarily converting analog processing to digital computation. In our society communicating is essential and radio communication systems play a fundamental role in enabling people to communicate (especially while on the move).

“A radio is a system that receives and transmits signals in the Radio Frequency (RF) part of the electromagnetic spectrum (ranging from 30 KHz to 300 GHz) in order to transmit and receive information.” Today radio communication systems are embedded in many devices commonly used in the everyday life, such as cellular phones, computers, and even vehicles. Until two decades ago, the only way to build a radio system was to use analog electronic techniques. The improvements in the Integrated Circuit (IC) technology, has enabled digital signal processing rather than analog signal processing in radio systems. The main idea behind a SDR system, is to realize a radio communication system where some or all of the physical layer functions are realized by software. In SDR we can replace the static analog platform with general purpose hardware that provides the waveform processing as implemented by software. The main advantages of SDR are its flexibility and portability i.e. upgrading a product is flexible and new features can be installed quickly, remotely, and without the need of any physical intervention.

2. Organization of paper

Section 4 of paper explains the basic principle of SDR, SDR Receiver and SDR Transmitter are elaborated in more detail. Section 5 explains SDR modules; Section 6 gives requirements for SDR Systems. Depending on the requirements various SDR platforms are explained in Section 7. Section 8 explains three case studies, which highlights how FPGA is more efficient and flexible platform for SDR. Section 9 gives the conclusion of the study and brief idea about future work and finally references are been enlisted.

3. Literature Review

In paper [4], authors reviewed the requirements for Software Defined Radio (SDR) systems for high speed wireless applications and compared the different technology choices available- from ASICs, FPGAs to digital signal processors (DSPs) and general purpose processors (GPPs). In paper [3], the authors proposed the use of hardware acceleration for processing waveform in SDR thus enhancing the output computation of the system. In 2004, the author [6] proposed the implementation of modulation chains for multi-standard communications on a dynamically and partially reconfigurable heterogeneous platform. All above papers have been elaborated in detail under case study section.

4. Software Defined Radio

A Software Defined Radio (SDR) is "a radio that is substantially defined in software and whose physical layer behaviour can be significantly altered through changes to its software" [1]. SDR can also be defined as a "Radio in which some or all of the physical layer functions are software defined"[8]. SDR is a radio system in which the waveform signal processing is performed digitally

A. Real SDR Model

In SDRs a large portion of the functionality is implemented through software. The Fig.1 shows the model of real SDR system. The antenna receives the analog radio signal. An intermediate step before conversion is needed in the receiver. This conversion to an intermediate frequency is required since SDRs must deal with radio frequency signals.

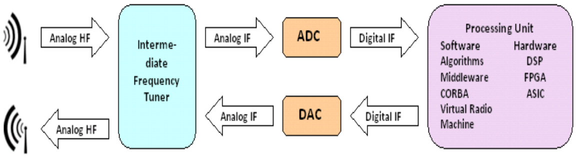

Fig.1: Real SDR Model

This step transforms the received high-frequency signal into a so called Intermediate Frequency (IF) by a tuner. Following this the IF is filtered and digitized. The filtering is done to prevent aliasing frequency signals into the band of frequencies that are being digitized. The stream is received and processed in a combination of software and hardware. [1] These hardware and software process the waveform. An output waveform is sent as a digital signal to be converted by DAC into an analog signal. A similar transformation can be made to shift the IF back for transmission. The analog signal is generally amplified and transmitted into air by a radio antenna

B. SDR Receiver

The fig.2 shows a block diagram of a software defined radio receiver. The RF tuner converts analog RF signals to analog IF. The A/D converter that follows digitizes the IF signal thereby converting it into digital samples. These samples are fed to the next stage which is the digital downconverter (DDC) shown within the dotted lines. The DDC is typically a single monolithic chip or FPGA IP, and it is a key part of the SDR system. The digital mixer and local oscillator translate the digital IF samples down to baseband. The FIR low pass filter limits the signal bandwidth and acts as a decimating low pass filter. The digital baseband samples are then fed to a block labelled DSP, which performs tasks such as demodulation, decoding and other processing tasks. [2].

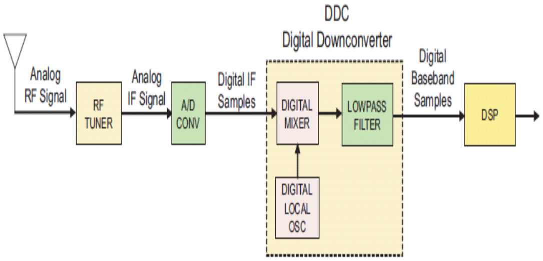

Fig.2: SDR Receiver

SDR Transmitter

The input to the transmit side of an SDR system is a digital baseband signal, typically generated by a DSP stage as shown in fig.3.The digital hardware block in the dotted lines is a DUC (digital upconverter) that translates the baseband signal to the IF. The D/A converter that follows convert the digital IF samples into the analog IF signal. Next, the RF upconverter converts the analog IF signal to RF frequencies.

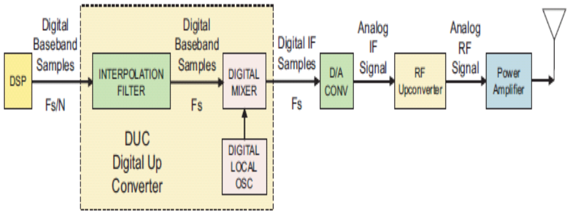

Fig.3: SDR Transmitter

Finally, the power amplifier boosts signal energy to the antenna. [2]

5. SDR Modules

The two-axis graph "Processing Intensity Vs. Flexibility", fig.4 shows some of popular signal processing tasks associated with SDR system. Processing intensity is the degree of highly repetitive operations.

The upper left area indicates dedicated functions like ADC & DDC that requires specialized hardware structures to complete the operations in real time. Flexibility defines how easily the functionality can be changed or customized for any specific applications

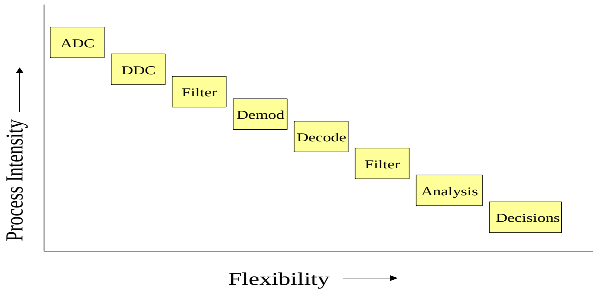

Fig.4: SDR Modules

The lower right area shows functions like analysis and decision making which are highly variable & often subjective. [2]

6. SDR System Requirements

The first natural requirement of software radio is flexibility, and configurability. The common trend between emerging wireless technologies is for higher data-rates, better quality of service, better security and adaptability and support for mobility. SDR System requirements are as following:

1. High Computation Capacity
2. Computation Accuracy
3. Memory Architecture
4. Communication Capacity at I/O Interfaces
5. Real-Time operation
6. Development Efforts and Flexibility

7. SDR Platforms

The main hardware alternatives that can be used to implement a SDR are DSP, GPP, ASIC, and FPGA. Broadly, the spectrum of digital hardware choices available for implementing a communication system range from the very specialized, inflexible but super-efficient Application Specific Integrated Circuits (ASICs) to highly programmable General Purpose Processors (GPPs) which sacrifice some cost, area and power efficiency. Between the two lie DSPs, FPGAs and many hybrid systems.

Table.1: Comparison of Platforms

	High-Speed DSPs	**Multiple ASICs**	**Parameter-ized Hardware**	**FPGAs**
Power Consumption	Very High	Very Low	Moderate	Low
Size	Modest	Large	Moderate	Low
Cost	Moderate/High	High	Moderate	Moderate /Low
Field Upgradable	High	None	Some	High
Silicon Evolution	Easy	Available	Moderate	Easy

Digital Signal Processors (DSPs) are microprocessors with architecture, instructions and features suited specifically for signal processing applications. Field Programming Gate Arrays (FPGA) is which contain DSP blocks that can be re-configured to work as parallel multiplier/adder or MAC. FPGA are extremely flexible and fast. [4]. The table.1 shows the comparison between DSP, ASICs, Parameterized Hardware and FPGAs

A. ASIC Platform for SDR

ASICs are chosen for ADC and DDC modules as these requires specialized hardware structures to complete the operations in real time.

B. GPPs/DSP Platform for SDR

Programmable GPP or DSPs are usually chosen for the analysis and decision making SDR modules because these modules are highly variable and often subjective. The fig.5 shows how various SDR modules can be implemented using ASIC and DSP. [2]

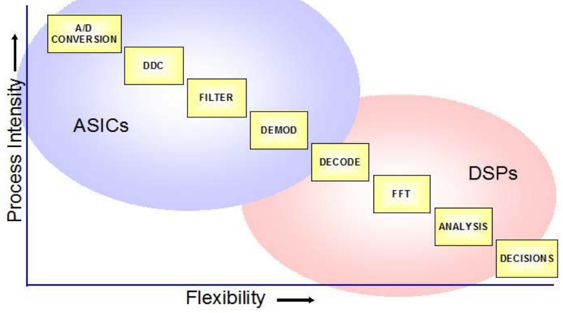

Fig.5: ASIC and DSP Platform for SDR

Their performance is increasing rapidly to provide the flexibility of DSP and the speed of ASICs [5]. FPGAs offer the advantages of parallel hardware to handle some of high process intensity functions like DDCs & the benefit of programmability to accommodate some of decoding and analysis functions of DSPs.

8. Case Studies

A. Software Radio: A Review of Design Considerations and Digital Hardware Choices

This case study reviews the requirements for Software Defined Radio (SDR) systems for high speed wireless applications and compares how well the different technology choices available- from ASICs, FPGAs to digital signal processors (DSPs) and general purpose processors (GPPs) - meet them. Which platform is the best depends very heavily on the target application.

Table.2: Comparison of different platforms for an SDR Implementation

	Freescale SC8144	Power6 (GPP)	Xilinx Virtex4-SX25
Operating Freq.	1 GHz	4.7 GHz	500 MHz
Computation Rate	16 GMAC/s	120GFlop/s (8 core)	256 GMAC/s
Numerical Precision	16-bit Fixed Point	64-bit Floating Pt.	Variable, Fixed Point
On-chip Memory	L1cache (I:16KB, D:32KB) L2 cache: 128 KB 10.5 MB Internal RAM	L1cache (I: 64KB, D: 64KB) L2 cache: 4MB	128 RAM blocks 18Kbits each
Memory bandwidth	N. A.	32 GB/sec	21 - 32 GB/s
Power Consumption	4.5 W	N.A.	4 W
Size	841 mm^2	341 mm^2	729 mm^2
Development ease	High	High	Low
Design flexibility	Medium	Medium	High

Table.2 shows a comparison of some commercially available platforms. All the platforms compared offer high computational capacity and can be considered for implementing a 4G-software radio system. [4]. Thus we can conclude from Table.2 that FPGAs offer more customization and cost-performance power efficiency than processors but at the cost of more complexity in system design.

B. The use of Hardware Acceleration in SDR Waveforms

The concept behind SDR is that more of the waveform processing can be implemented in reprogrammable digital hardware so a single platform can be used for multiple waveforms. Most SDR systems utilize a general purpose processor (GPP), digital signal processor (DSP), and FPGA in their architectures. Moreover, with general purpose routing resources available in the FPGA, hardware acceleration units can run in parallel to further enhance the total computational output of the system. This case study demonstrates the use of FPGA for Hardware Acceleration [3].

Hardware Acceleration Techniques are:

i) Core Processors and Custom Instructions

The core processor can be “hard-core” processor, which are a physical part of the FPGA silicon, or “soft core” processor, processors, which are IP blocks, downloaded as part of the design running on the FPGA. Advantage of softcore over hard core is they are infinitely flexible.

Custom instructions are algorithm-specific additions of hardware to the soft-core microprocessor’s arithmetic logic unit (ALU). These new hardware instructions are used in place of a time-critical piece of an algorithm, recasting the software algorithm into a hardware block. The most efficient use of custom instruction occurs when the algorithm to be accelerated is a relatively atomic operation that is called often and operates on data stored in local registers. For example with soft-core processors running on an FPGA, the floating point functions can be implemented as custom instructions extending a soft-core microprocessor's ALU. The performance improvement of these hardware custom instructions over their software counterparts can be dramatic.

ii) Hardware acceleration co-processors

Hardware-acceleration co-processors can be used to accelerate processors or DSPs. Custom instructions differ from hardware acceleration co-processors in that custom-instructions are an extension of an ALU which is relegated to a soft-core microprocessor. One of the key advantages of hardware acceleration coprocessor is that it is wrapped in a DMA so it has direct access to memory i.e. the coprocessor can work on a block of memory without intervention from the processor. Situations where hardware acceleration coprocessors could be used over a custom instruction are:

- Algorithms do not only use register variables (non atomic).
- Operations are more complex (often a subroutine in software).
- Transformation of data is done on a large data block

iii) Application Specific Instruction set Processors

Application specific instruction-set processors (ASIPs) are a special case of the hardware acceleration co-processors. An ASIP combines the flexibility of a software approach with the efficiency and performance of dedicated hardware. An ASIP is a processor that has been targeted to perform a specific task or set of related tasks. Software defined radios implement algorithms in software to improve portability, lifetime costs and retargetability. However, achieving cost and performance requirements necessitates the use of application specific hardware.

ASIPs on an FPGA are composed of smaller building blocks that can be reconfigured on the fly to implement more than one higher level function. For example implementation of Fast Fourier Transform (FFT) blocks and Finite Impulse Response (FIR) filters. These two high-level algorithms share many common sub blocks. By changing the interconnect between these sub blocks the ASIP can be altered to implement the FFT instead of the FIR in hardware.

Thus the FPGAs can be used for hardware acceleration, through custom instructions added to soft-core processors, hardware acceleration co-processors enhancing the processing power of GPPs and DSPs, or application specific instruction set processors providing tight, efficient reconfigurable building blocks for computation, offers promising architectural options that are helping to make SDRs a reality.

C. Software Radio and Dynamic Reconfiguration on DSP/FPGA Platform

This case study discusses the implementation of modulation chains for multi-standard communications on a dynamically and partially reconfigurable heterogeneous platform. DSP/FPGA platform is more advantageous instead of a multi DSP platform since the FPGA supports efficiently intensive computation components, which reduces the DSP load. DSP/FPGA Sundance Platform is used for implementation. The design is implemented in MATLAB (for DSP) and in VHDL (for FPGA).

Furthermore, partial dynamic reconfiguration increases the overall performance as compared to total dynamic reconfiguration since there is 45% of bit stream size reduction, which leads to a 45% decrease of the whole reconfiguration time. The implementation of modulation chains for multi-standard communications proves the availability of new technology to support efficiently Software Defined Radio [6].

7. Conclusions and Future Work

Software-defined radio is an adaptive, future-proof solution to making wireless networks highly flexible. The variety of technology choices available for implementing SDR systems range from ASICs, FPGAs, general and special purpose processors and everything in between. However, FPGA is more advantageous as compared to other platforms as it provides reconfiguration, dynamic partial reconfiguration, parallel processing, hardware multipliers for DSP, flexible memory structures, parallel and pipelined dataflow, flexible I/O, high speed, IP cores available for SDR. FPGA also provides hardware acceleration to enhance the efficiency of the SDR system. Thus FPGA is a better platform for implementation of SDR modules.

The next goal of the work is to demonstrate a practical design and implementation of PSK modulation and demodulation technique on FPGA. The work will involve the design and implementation of the BPSK, QPSK and 8-PSK modulation and demodulation schemes on FPGA. The PSK modulator and demodulator algorithms will be simulated using MATLAB R2012a and implemented on FPGA using Xilinx ISE 14.2 System Generator.

References

[1] Carlorinaldi, "Performance evaluation and optimization of an OMAP platform for embedded SDR Systems.
[2] Rodger H. Hosking "Software Defined Radio Handbook", Eighth Edition.
[3] David Lau, Jarrod Blackburn, Joel A. Seely, "The use of Hardware Acceleration in SDR Waveforms", Altera Corporation.
[4] Parul Gupta, Shivkumar Kalyanaraman, Pradipta Dey, Qing Wangz, Jian Wen Chenz, Yong Hua Lin "Software Radio: A Review of Design Considerations and Digital Hardware Choices", IBM Research Lab.
[5] SDR Forum "SDRF Cognitive Radio Definitions" Working Document SDRF-06-R-0011-V1.0.0, November 2007.
[6] [Online] J. P. Delahaye, G. Gogniat, C. Roland, P. Bomel"Software radio and Dynamic Reconfiguration on a dsp/fpga platform" IETR/Supelec - Campus de Rennes, Av. de la Boulais, BP 81127.
[7] (Online)www.site.uottawa.ca/~mbolic/elg6163/SDR.doc.
[8] (Online) www.sdrforum.org .

Archana Lalge received her B.E. degree from University of Pune in 2001. She is currently pursuing M.E in VLSI and Embedded Systems from P.C.C.O.E, Pune University. She has 7 years of teaching experience.

Mayuri Karpe received her B.E degree from Pune University in 2010. She is currently pursuing ME in VLSI and Embedded systems from P.C.C.O.E Pune University. She is currently working in LSI India Pvt. Ltd as Intern in DFT (Design for Testability) area.

Sheetal Bhandari is an assistant professor of electronics and telecommunication engineering at Pimpri Chinchwad College of Engineering, University of Pune, India. She received her B.E. and M.E. degree from University of Pune in 1998 and 2006 respectively. She has completed her PhD in the area of Reconfigurable Computing in 2013. She is been teaching for about 8 years and has entrepreneurial stint of 4 years. Her academic focus is on Microelectronics and VLSI Design. Her research interests include Partial Reconfiguration and HW-SW Co-Design.