An FPGA-Based Software Defined Radio Platform for the 2.4GHz ISM Band

Antonio Di Stefano, Giuseppe Fiscelli, Costantino G. Giaconia
Dipartimento di Ingegneria Elettrica, Elettronica e delle Telecomunicazioni, Università degli Studi di Palermo
Viale delle Scienze, ed. 9 – 90128 Palermo, Italy
distefano@diepa.unipa.it, giuseppefiscelli@gmail.com, costantino.giaconia@unipa.it

***Abstract*—A prototype of a Software Defined Radio (SDR) platform has been successfully designed and tested implementing a reconfigurable IEEE 802.11 and ZigBee receiver. The system exploits the reconfiguration capability of an FPGA for implementing a number of receiver configurations that share the same RF front-end. Configurations can be switched at run time, or can share the available logic and radio resource.**

I. INTRODUCTION

Software Defined Radio (SDR) [1] is the ability of changing the characteristics of a transmitting and receiving radio device without physically modifying the hardware. This implies the possibility of changing coding scheme, modulation, bandwidth and channel access techniques. This can be done by using a suitable RF stage and a reprogrammable or reconfigurable hardware in the baseband processing and medium access control sections. An implementation of such a system can be really challenging, since all the described elements are very complex to design and implement, and usually require a great quantity of logic resources or extremely fast (and costly) Digital Signal Processors. For this reason only a reconfigurable fabric (in the widest connotation of the term) and a dedicated implementation can achieve the result.

In this work we present a first attempt to realize an SDR platform suitable for implementing and testing reconfigurable wireless architectures working in the unlicensed 2.4GHz Industrial, Scientific, and Medical (ISM) band. The proposed system employs a commercial 802.11b RF stage and an FPGA used to implement different receivers baseband and Medium Access Control (MAC) architectures, and to experiment different solution to the reconfiguration problems. The system can also be used as a low level channel sniffer to deeply analyze interactions between the MAC and Physical (PHY) level, and protocol interoperability and compliance issues for different standards [2]. As will be described in the following, in order to test the feasibility of this approach an IEEE 802.11 [3] and an IEEE 802.15.4 [4] (also known as "ZigBee") receiver were implemented.

The two receivers can work concurrently or can be loaded (through reconfiguration) at run time. As can be seen from Fig. 1 the 802.11b channel has a bandwidth of about 22MHz, while the ZigBee signal occupies a bandwidth of about 2MHz and has far less stringent requirements in term of RF stage linearity, noise and sensitivity. For this reason the 802.11b RF front-end can be used for both the standard.

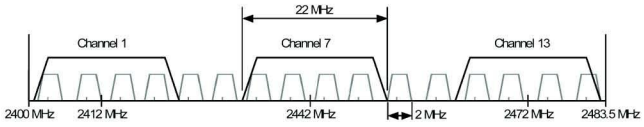

Figure 1. The three 802.11 and the sixteen 802.15.4 orthogonal channel and their respective allocation and bandwidth.

The following sections (from II to IV) will describe in detail the system hardware, architecture and implementation of the two receiver respectively. Section V presents the implementation results, and finally in Section VI some conclusions are drown.

II. SDR PLATFORM HARDWARE

The SDR platform is composed by a 2.4GHz RF chain, a Field Programmable Gate Array, capable of implementing either a BaseBand Processor or some test an measurement instruments, and an host computer used to control and configure the whole system (Fig. 2).

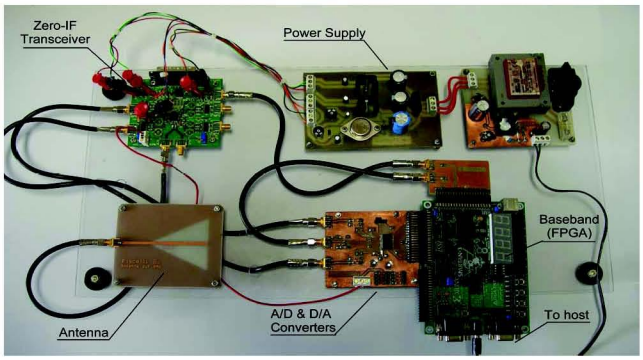

Figure 2. Picture of the SDR prototype.

The RF front-end is based on a MAX2820 802.11b Zero-IF transceiver from Maxim [5] that integrates a complete 2.4GHz transmission and reception path from RF to baseband and vice versa (Voltage Controller Oscillator, Integer-N Phase Locked Loops, Low Noise Amplifier, direct down and up-converter, and filters). The RF signal, coming from an omnidirectional isosceles triangular slot antenna [6], designed for the 2.4GHz ISM band, is amplified and down-converted by a quadrature demodulator, thus obtaining the two analog In-Phase and Quadrature (I&Q) signals. The two baseband signals are then provided to the FPGA using two 6 bit matched Analog to Digital converters operating at 22Msps. This frequency allows to get two samples per chip in case of 802.11 signals and 11 samples per chip for ZigBee signals.

The FPGA used is a mid density SRAM-based device from Xilinx that is used to implement a reconfigurable baseband processor/MAC and/or custom measurement instruments. The board has also 1MB of external SRAM with 10ns access time, that can be used to store sampled data, test vector or measures. The FPGA also controls the RF section through dedicated programming lines, the Automatic Gain Control (AGC) input and transmit power control. Data output from the FPGA are available to an external MAC or can be directly sent to a host PC. The host is also used to reconfigure the FPGA in order to implement the desired functionality.

III. 802.11 Receiver Architecture

The IEEE 802.11 standard (1997 edition) employs a Differential Binary Phase Shift Keying (DBPSK) and a Differential Quaternary Phase Shift Keying (DQPSK) to modulate signals with a bit rate of 1Mbps and 2Mbps respectively. In both cases the Direct Sequence Spread Spectrum (DSSS) technique is applied using an 11 chip Barker code (i.e. 10110111000) as a spreading sequence. Every frame preamble is transmitted using the 1Mbps rate and modulation, while the 2Mbps rate can be used for data field. A scrambling process is also applied to data before the modulation process.

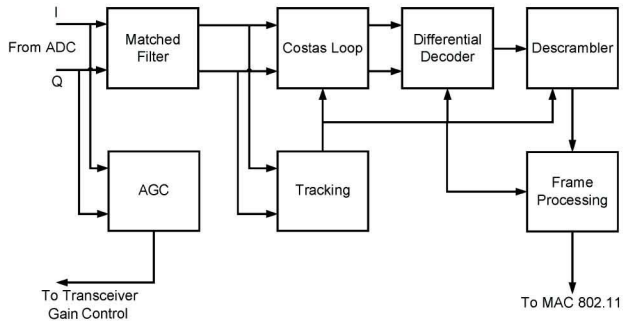

Figure 3. The 802.11 baseband architecture.

Considering the above mentioned specifications, it is clear that demodulating 802.11 signals is not a trivial task, requiring a quite complex receiver. Fig. 3 shows the block diagram of the implemented architecture. Since the received signals are characterized by very wide variations of amplitude (in the range of 50dB) an Automatic Gain Control was implemented [7]. The circuit works by evaluating the magnitude of the complex signal and applying to it a low-pass filter. The AGC output signal is then fed to the RF stage through a 4 bit D/A converter. An additional 1 bit signal is use to enable the high gain mode of the LNA when the input signal is below a certain threshold.

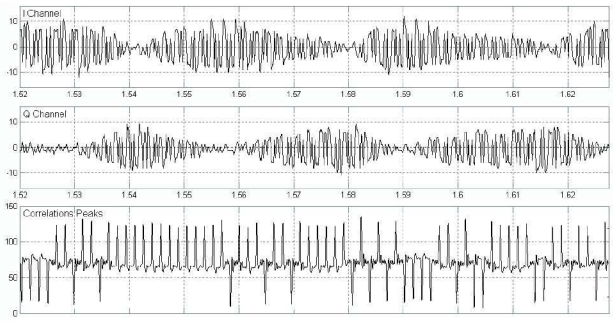

Figure 4. An 802.11 BPSK baseband complex signal and the correlation peaks obtained using the 11 chip Barker code. The visible envelope in the baseband signals is the effect of the carrier frequency offset.

The I and Q signals are firstly processed by a channel matched filter, implemented as a sliding correlator with the Barker sequence. This step allows to de-spread the signal and to attenuate the effects of channel noise, multipath fading and other propagation issues. Moreover, the excellent autocorrelation properties of the Barker sequence makes easier to extract symbol timings and to remove the frequency and phase carrier offset. This is possible since the correlation produce a sharp peak for each received symbol (i.e. data bit). This is shown in Fig. 4. The bit synchronization signal is obtained by tracking the peak positions, that is detected by finding the maximum value among 22 samples (this method does not require to set an explicit threshold level).

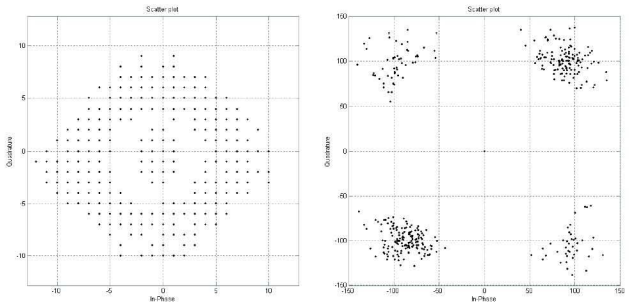

Figure 5. An incoming 802.11 2Mbps complex signal before and after correlation and de-rotation. The points on the first and third quadrant show a greater density because of the BPSK modulated preamble.

In order to remove the carrier frequency and phase offset an hard limited Costas loop is used [8]. The loop must be capable of switching, while receiving a frame, between a BPSK (1Mbps data and preambles) and a QPSK signal (2Mbps data). This is obtaining controlling the loop error

function. The Costas loop is preceded by a 22:1 downsampler synchronized with the bit timing reference signal. This makes the loop behavior much more robust with respect to noise, since it will operate mainly on significant data samples. As can be seen from Fig. 5 this process is quite effective, since a very noisy 2Mbps signal has been successfully de-rotated.

After this process a differential decoding removes the remaining 90° or 180° phase ambiguity. Digital data are then descrambled and decoded, and the CCITT CRC-16 is computed over the data belonging to the physical header. Finally data are passed to the MAC or to an external host for further processing.

IV. ZigBee Receiver Architecture

The IEEE 802.15.4 standard employs an Offset-QPSK (O-QPSK) modulation with an half sine pulse shaping (similar to the Minimum Shift Keying modulation). A DSSS spreading is obtained by using 16 quasi-orthogonal codes, each composed by 32 chip and encoding 4 data bits (see [4]). The chipping rate is 2Mcps and the data rate obtained is 250Kbps. Each 32 bit code is divided in two 16 bit sub-codes that are separately modulated and mapped to the I and Q channel (Fig. 6).

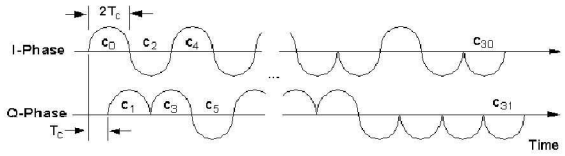

Figure 6. ZigBee O-QPSK modulation scheme.

A number of receiver architecture has been proposed to demodulate signals with similar characteristics [9]. However these usually need to perform expensive processing steps (in term of silicon area and number of gates) requiring for example a pulse shape matched filter, and a carrier recovery loop for the coherent demodulation. The signals obtained have to be yet de-spreaded using 16 parallel 32 chip correlators.

In order to make the implementation as compact and efficient as possible, a dedicated architecture was used. This is based on a non-coherent receiver algorithm and a special correlation and synchronization block.

This technique exploits the phase continuity of an O-QPSK signal with half sine pulse shaping. Tracking the phase change of the received signal it is possible to obtain information about the transmitted chips and their timing. In particular the circuit follows the direction of rotation of the incoming signal vector (I+jQ), and its changes using the following expression, derived from the Costas loop error term for QPSK signals:

$$\Delta\theta = \frac{\partial I}{\partial t}\cdot \mathrm{sgn}(Q) - \frac{\partial Q}{\partial t}\cdot \mathrm{sgn}(I)$$

The phase increment signal ($\Delta\theta$, shown in Fig. 7) could be used to reconstruct the transmitted chips, while its zero crossings can be used as a timing reference (the minimum period between two zero crossing is half the chip time). In this case however both these steps are not necessary. In fact, instead of correlating the received chips with the spreading codes, it is possible to generate a new set of codes that can be directly used with the $\Delta\theta$ signal (Table I).

TABLE I. Spreading Codes Used for $\Delta\Theta$ Signal

Data Symbols	Chip values (c0, c1… c31)
0000	**0011**11110001000010100011001001**10**
0010	0110**0011**1111000100001010001100**10**
0100	00100110**0011**1111000100001010001**1**
0110	0011001001**10011**1111000100001**010**
1000	10100011001001**10011**1110001**0000**
1010	0000101000110010011**0011**1111**0001**
1100	0001000010100011001001**10011**111**1**
1110	1111000100001010001100100**1100011**
0001	**1100**000011101111010111001101**1001**
0011	1001**1100**000011101111010111001**101**
0101	1101100**1100**000011101111010111**00**
0111	1100110110011**1100**000011101111**0101**
1001	0101110011011001**1100**000011101**111**
1011	1111010111001101100**1100**00001**110**
1101	1110111101011100110110**011000**0000
1111	0000111011110101110011011001**1100**

It has been found that these codes retain the properties of the original codes, i.e. they can be obtained from a single code through rotation and inversion. These properties simplifies the realization of the correlator block, that can be implemented as a bit correlator in which all the 16 code words (obtained through a bit rotation) are serially checked within a code period (i.e. 32 chip). The code found to have the maximum correlation value is selected by a biggest picker. The outputs of this process are the decoded 4 data bits (Fig. 8). The correlator is also used to acquire the timing and phase reference during the reception of each frame preamble. These information are provided to the following processing element.

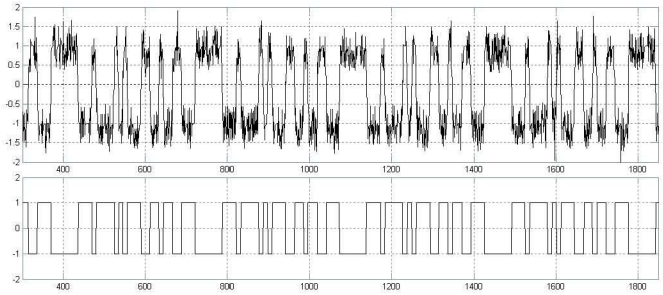

Figure 7. The upper figure shows the $\Delta\theta$ signal obtained for an incoming ZigBee signal. This periodical pattern refers to the sequence of four "0x00" bytes of the preamble that is used for bit timing synchronization.

The last block is represented by the frame processing logic, that decodes the data fields to send them to the MAC

or host. A schematic overview of the ZigBee baseband is showed in Fig. 9.

It has to be noted that the first processing element consists in an 1MHz low-pass filter used to reduces signal noise and co-channel interferences.

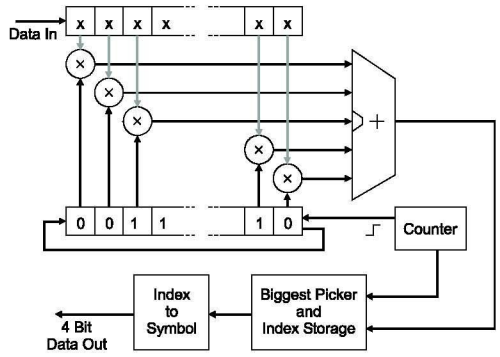

Figure 8. ZigBee bit correlator and symbol evaluation block.

V. BASEBAND PROCESSORS IMPLEMENTATION

All the digital signal processing elements required to demodulate the 802.11 and the ZigBee signals were implemented into the FPGA. The two baseband architectures were designed using Matlab and Simulink for system level simulation and optimization, and then ANSI C, VHDL and ModelSim for the RTL description, simulation and implementation. For both architecture a 6 bit fixed point representation was used.

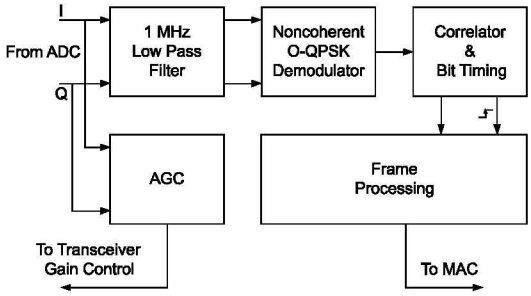

Figure 9. ZigBee baseband architecture.

The design was implemented in a Xilinx Spartan3-200 FPGA [10], obtaining the following occupation: the 802.11 configuration required about 613 slices of 1920 (31%) plus 4 embedded multipliers (6x6 bit), while the ZigBee configuration required less than 200 slices (about 10%). The sine and cosine lookup table for the Costas loop used 64 values (6 bit) to encode a quarter of a sinusoid. The master clock used is 44MHz and 22MHz for the 802.11 and the ZigBee configuration respectively. The 22MHz frequency was directly used in both configurations as sampling frequency, since both the architectures have no need for sample synchronization. Thanks to their modest resource occupation, both configurations can reside inside the FPGA and work concurrently sharing the same RF path, but a dynamic reconfiguration can be also used to switch between the two. In this case the configuration switch is done by a software running on the host PC that sends the chosen bitstream through the parallel port when desired. A complete reconfiguration can be carried out in less than 3 seconds, but this time can be dramatically reduced (to some milliseconds) using other configuration methods, such as partial configuration or slave parallel download. More research has to be done aimed to exploit the partial and run time reconfiguration possibility of the FPGA [11]. This will allow extremely short context switch among different standards and a more efficient resource utilization.

VI. CONCLUSIONS

A fully functional prototype of a Software Defined Radio has been realized. The platform has been successfully used to implement a reconfigurable 802.11/ZigBee receiver. Thanks to its deep flexibility the platform has been also used to test new reconfigurable wireless receiver architectures, and to perform a low level monitoring of the 2.4GHz ISM band addressing protocol interoperability and compliance issues. Other modulation schemes (such as Cypress WirelessUSB® and Bluetooth) can be easily implemented as well and integrated with a MAC processor, further expanding the possibilities. Future studies will be directed toward the use of partial reconfiguration and the use of modular and configurable processing elements.

REFERENCES

[1] W. H. W. Tuttlebee: "Software-Defined Radio: Facets of a Developing Technology", IEEE Personal Communications, April 1999.

[2] A. Di Stefano, et al., "On the Fidelity of IEEE 802.11 Commercial Cards", IEEE Wireless Internet Conference 2005, July 2005, Budapest.

[3] IEEE Std 802.11, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Institute of Electrical and Electronic Engineers, November, 1997.

[4] IEEE Std 802.15.4, Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs), Institute of Electrical and Electronic Engineers, October, 2003.

[5] Maxim Co. "MAX2820 - 2.4GHz 802.11b Zero-IF Transceivers" datasheet, Jan. 2004.

[6] W. S. Chen, F. M. Hsieh: "A Broadband Design for a Printed Isosceles Triangular Slot Antenna for Wireless Communications", Microwave Journal, July 2005.

[7] D. N. Green, "Lock-In, Tracking, And Acquisition of AGC-Aided Phase-Locked Loops", IEEE Transactions On Circuits And System, Vol. Cas-32, No. 6, June, 1985.

[8] M. K. Simon, "Tracking Performance of Costas Loop with Hard-Limited In-Phase Channel", IEEE Trans. on communications, Vol. Com-26, April, 1978.

[9] Pasupathy, S.: "Minimum shift keying: A spectrally efficient modulation", IEEE Communications Magazine, vol. 17, issue 4, Jul. 1979.

[10] Xilinx Co., "Spartan-3 FPGA Family: Functional Description", datasheet, Aug. 2004.

[11] Xilinx Corp., "Two Flows for Partial Reconfiguration: Module Based or Difference Based", application note, Sept. 2004.