SERBIAN JOURNAL OF ELECTRICAL ENGINEERING
Vol. 1, No. 3, November 2004, 125 - 136

FPGA-based Prototyping of IEEE 802.11a Baseband Processor

Dejan M. Dramicanin [1], Dejan Rakic [1], Slobodan Denic [1], Veljko Vlahovic [1]

Abstract: In technical literature and especially in domestic, predominant way to examine performance of 802.11a-based systems are experiments in simulations. In this paper, we present FPGA based 802.11a prototype, which gave us a possibility to gain closer insight into the problems of OFDM system implementation. A specific design of baseband modem physical layer is discussed, along with the presentation of the FPGA prototyping platform on which it was developed. Prototype is implemented on the latest generation of FPGA chips, using state-of-the-art tools for DSP development. Custom made development environment, and design flow optimised for rapid prototyping of software defined radios, are also presented in the paper.

Keywords: Processor, FPGA.

1 Introduction and Outline

IEEE 802.11a Wireless LAN standard is one of the most recent advances in the field of commercially applied digital communications. The 802.11a system is based on OFDM modulation, with the packet structure and MAC layer definition optimised for wireless transmission over the slowly changing multipath channel. In-depth knowledge of the solutions applied in implementation of 802.11a modem is ideal starting point for mastering the usage of OFDM – digital modulation with the greatest potential to be found in communicators of the next generation.

This article presents 802.11a baseband processor prototype, implemented on FPGA platform. Latest generation of FPGA have extraordinary performance, vast array of configurable logic tiles and built-in system blocks (multipliers, RAM, clock management units, general purpose processors), and have configurable I/O that can easily fit into virtually any digital hardware environment. This technology provides the signal-processing engineer with the ability to construct a custom data path that is tailored to the application at hand. FPGAs offer the flexibility of instruction set digital signal processors while providing the processing power and flexibility of an application-specific integrated circuit (ASIC). FPGA have given new dimension to research and development of software defined radios. Configurable arrays are inevitable in the process of making

[1] Signum Concepts Inc.

functional prototype for proving the theoretical concepts in the interaction with realistic exploitation environment. Utmost advantage given by the flexibility of FPGA during the development process is that functional prototype is designed in the cost effective and time effective manner. One of the motivations of the text that follows is to show that it is possible to establish development laboratories with minimal investments, for research and development of psychical layer in tomorrow's digital communications. This could achievable even in our country where financial constraints post us far behind the cutting edge.

Paper is organized as follows. The next section defines the structure of the 802.11a baseband processor, suggesting the techniques to be used for critical processing tasks: packet detection, timing acquisition, frequency acquisition, channel estimation, and post-FFT processing (tracking). These techniques are described at a glance. The second part presents FPGA platform and the development environment used in this project. Custom tailored FPGA design flow is presented, optimised for rapid development of software defined radios in small teams working off the site on the outsourced projects. This design flow relies on SW/HW framework based on virtual instrumentation concepts. Key points of the framework are also presented.

2 Structure of 802.11a Prototype

Fig. 1 shows block diagram of 802.11a modem prototype

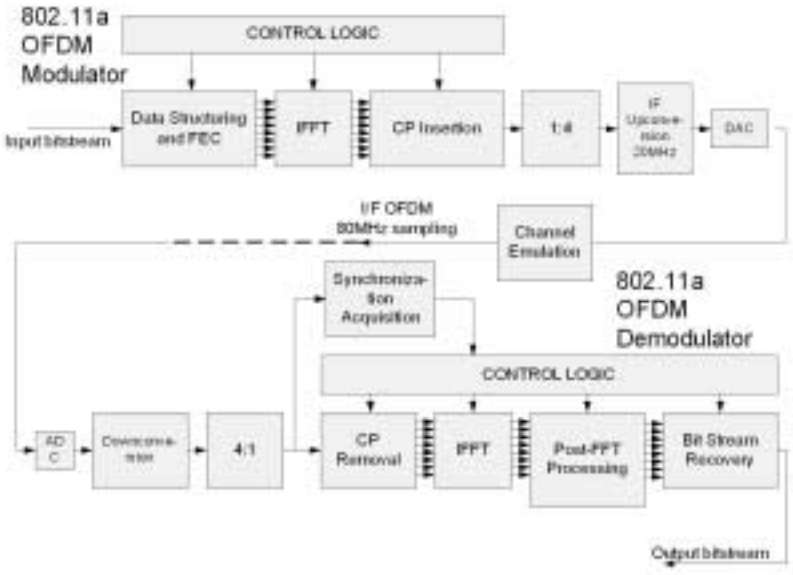

Fig. 1 - *802.11a prototype block diagram.*

Design is split on two separate hardware boards, one with modulator and the other with demodulator. Sampling frequency in 802.11a baseband 20 Msps [1]. In the system

on Fig. 1, sampling time is increased by the factor of 4 ($f_S = 80$ Msps), baseband signal is upconverted on $f_S / 4$ centre frequency, and transmitted through the channel as a real signal.

2.1 Synchronization preamble structure

Fig. 2 shows the structure of 802.11a synchronization preamble [1].

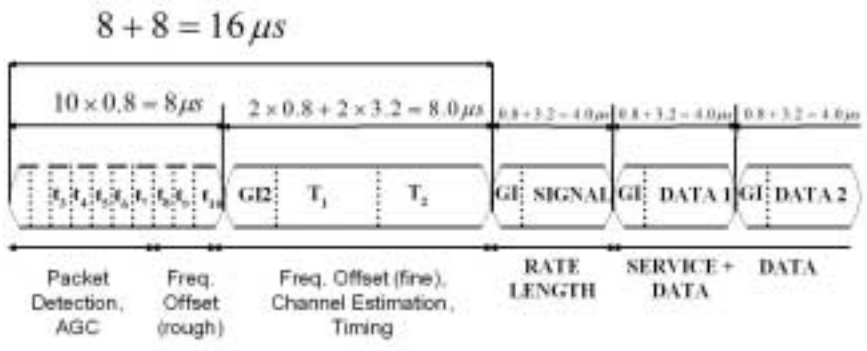

Fig. 2 - *802.11a synchronization preamble.*

First section consists of 10 identical short symbols with 16 samples each (with 200 Msps sampling frequency). Short symbol burst is followed by two long symbols, with 64 samples, with long cyclic prefix of 32 samples. Three standard symbols, each with 64 samples and 16 samples of cyclic prefix, are within synchronization preamble, but they are not of interest for synchronization on physical layer.

2.2 Packet Detection

Packet detection processing recognizes the presence of the signal on the input of the receiver. We suggest method based on two sliding auto-correlations shifted in time [2]:

$$a_n = \sum_{k=0}^{N-1} r_{n-k} r_{n-k}^* = \sum_{k=0}^{N-1} \left| r_{n-k} \right|^2 \quad \text{and} \tag{1}$$

$$b_n = \sum_{k=0}^{N-1} r_{n+k} r_{n+k}^* = \sum_{k=0}^{N-1} \left| r_{n+k} \right|^2 . \tag{2}$$

Variable $m_n = a_n / b_n$ peaks when auto correlation a_n measured the signal with noise, and b_n only noise. Algorithm is illustrated on the Fig. 3.

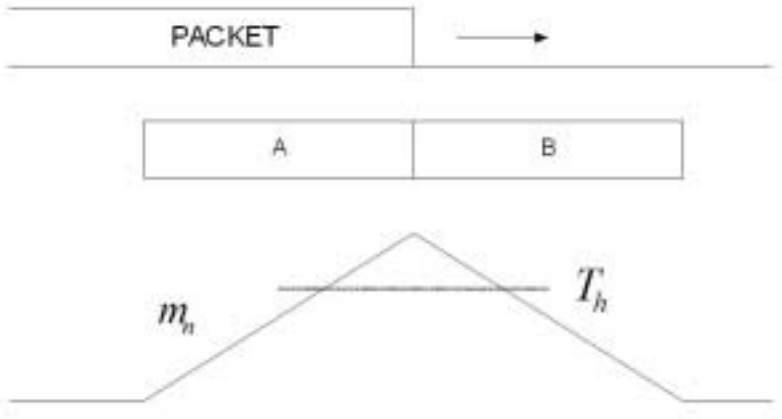

Fig. 3 - *Packet detection with sliding auto correlations.*

In the peak point, m_n has value

$$\max(m_n) = \frac{S+N}{N} = \frac{\boldsymbol{S}}{\boldsymbol{N}} + 1, \tag{3}$$

so the detection threshold could be set according to expected SNR. (SNR value detected as a result of this processing can be used later in channel estimation algorithms). Efficient algorithm implementation is shown on the Fig. 4.

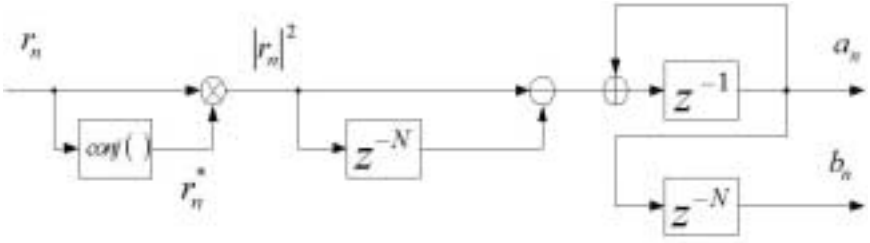

Fig. 4 - *Sliding auto correlation algorithm implementation.*

2.3 Timing Acquisition

When the packed is detected, timing acquisition algorithm is invoked. The task is to align frames for FFT processing. Acquisition is also based on auto correlation

$$M(n) = \sum_{m=0}^{143} y^*(n+m)\,y(n+m+M)$$
$$n' = \arg(\max(M(n))) \quad . \tag{4}$$

The auto correlation function $M(n)$ has maximum on the beginning of cyclic prefix of long symbols. Physical implementation of this processing unit is shown on the Fig. 6.

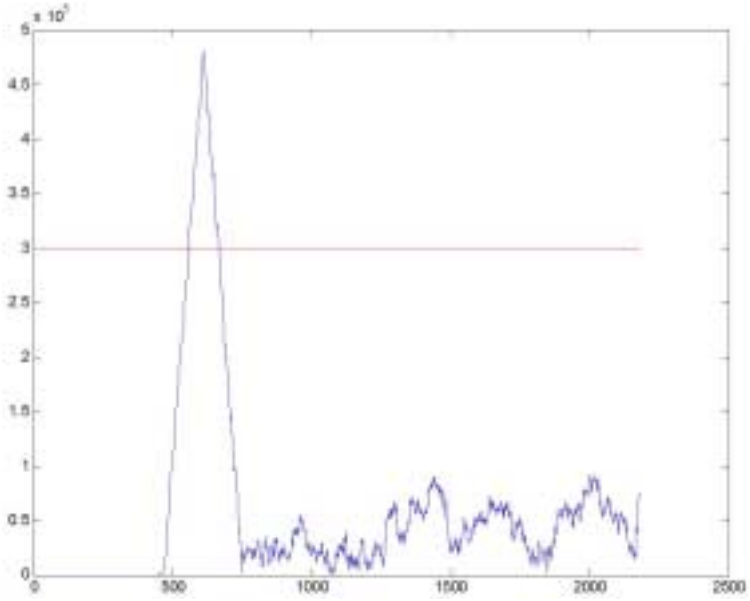

Fig. 5 - *Timing acquisition.*

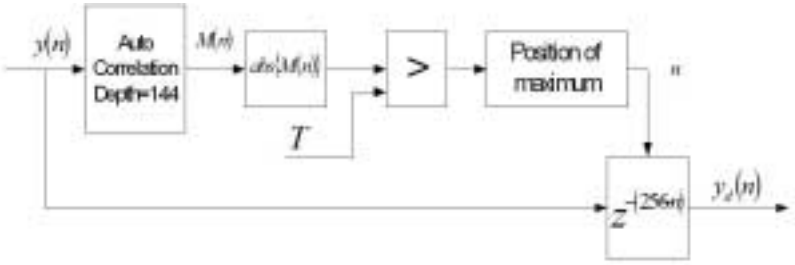

Fig. 6 - *Timing acquisition algorithm implementation.*

2.4 Frequency Offset Acquisition

In the presence of frequency offset, transmitted signal, $y(n)$,

$$y_c(n) = y(n)\mathrm{e}^{\mathrm{j}2\pi f_0 \frac{n}{N}}. \qquad (5)$$

For the rough estimate of frequency offset, auto correlation on short symbols is applied [3],

$$K = \sum_{n=0}^{15} y_c^*(n) y_c(n+16) = \mathrm{e}^{-\mathrm{j}2\pi \frac{f_{est}}{4}} \sum_{n=0}^{15} |y_c(n)|^2 . \tag{6}$$

Phase accumulation is proportional to frequency offset. Largest offset that could be determined without ambiguity is $\pm 2\Delta f$ (Δf stands for frequency distance between two OFDM carriers, in the case of 802.11a this is 312.5 kHz), so this information is used to roughly determine the frequency offset,

$$f_K = \begin{cases} 0 & -\frac{\pi}{4} \le \arg(K^*) \le \frac{\pi}{4} \\ \frac{\pi}{4*64} & \frac{\pi}{4} < \arg(K^*) \le \frac{\pi}{2} \\ \frac{\pi}{2*64} & \frac{\pi}{2} < \arg(K^*) \le \frac{3\pi}{4} \\ \frac{3\pi}{4*64} & \frac{3\pi}{4} < \arg(K^*) \le \pi \\ -\frac{\pi}{4*64} & -\frac{\pi}{2} \le \arg(K^*) < -\frac{\pi}{4} \\ -\frac{\pi}{2*64} & -\frac{3\pi}{4} \le \arg(K^*) < -\frac{\pi}{2} \\ -\frac{3\pi}{4*64} & \pi \le \arg(K^*) < -\frac{3\pi}{4} \end{cases} . \tag{7}$$

(12)

For fine frequency offset recovery, auto correlation on long symbols is used, with the frequency span of $\pm 0.5\Delta f$,

$$J = \sum_{n=0}^{63} y_c^*(n) y_c(n+64) . \tag{8}$$

Finally, frequency offset estimate is given with

$$f_{est} = f_K + f_J = f_K + \frac{1}{128\pi} \arg \frac{J^*}{|J|} . \tag{9}$$

Implementation of frequency recovery circuit is shown on Fig. 7.

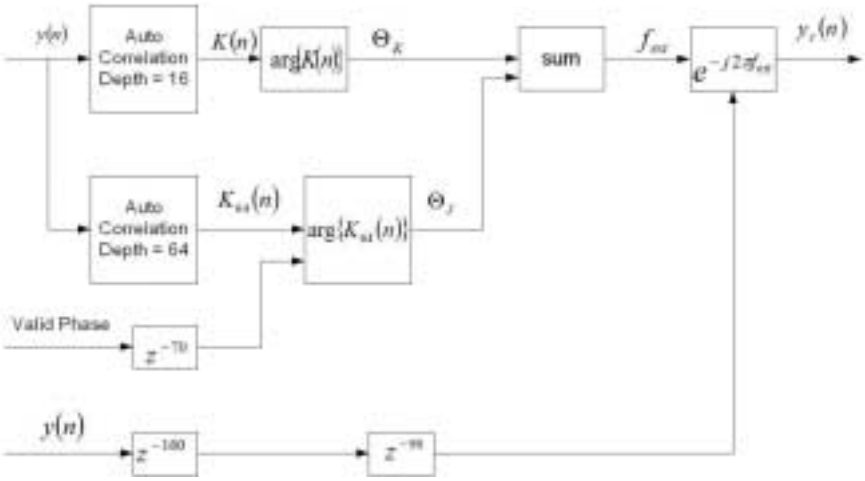

Fig. 7 - *Frequency offset acquisition algorithm implementation.*

2.5 Channel Estimation

Channel influence on transmitted signal is modelled with

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{n}\,, \tag{10}$$

where $\boldsymbol{H} = diag\{h_0 \quad h_1 \cdots h_{N-1}\}$ diagonal matrix consists of N-point DFT of channel impulse response $\delta(n)$.

Applying least squares (LS) criterion, channel is estimated as [3]

$$\hat{h}_{LS1}(k) = \frac{y(k)}{x(k)}, \quad k \in P\,, \tag{11}$$

where $y(k)$ is received symbol, $x(k)$ known pilot symbol (in this case this is the long symbol in synchronization preamble), and $P = \{1,\ldots,26,38,\ldots,63\}$ is a subset of active OFDM carriers.

2.6 Post-FFT Processing

Post-FFT processing has two chores: channel equalization, and timing and frequency synchronization tracking. Equalization is performed with frequency domain vector multiplication of received frame with compensation vector acquired in the process of channel estimation (11).

Because of residual frequency offset Δf, and because of difference in sampling frequencies of two boards, ξ, after every symbol the k^{th} OFDM carrier will receive additional phase rotation [4]

$$\Delta\varphi(k) = b + ak = 2\pi T_S \Delta f + 2\pi \frac{T_S}{T_u} \zeta k \ . \tag{12}$$

Frequency offset and timing tracking is performed by analysis of four pilots assigned to every OFDM data frame. Based on $\Delta\varphi$ of pilots, parameters a and b are determined, and presented to equalization block which corrects the phases of the other carriers.

3 Development Environment

3.1 FPGA Platform

Prototype is developed on *Nallatech ExtremeDSP* boards. This system has modular architecture, where add-on cards are inserted into the motherboard's DIME-II compatible slots [5]. *ExtremeDSP* package consists of motherboard with one DIME-II slot (*BenONE*) and add-in card with FPGA chip, two input and two output channels (*BenADDA*).

Basic tasks of the motherboard are to control configuration chain of FPGA chip on add-in card, as well as to provide communication path between host PC and user design. Supported interfaces are PCI 64/33 and USB 1.1. Add-in card in this set is equipped with *Xilinx Virtex-2 XC2V2000-4* FPGA chip. Two digital-to-analog converters has 160 Msps maximum sampling rate, and analog-to-digital converters have 105 Msps with analog bandwidth of 300 MHz .

Software bundle is exceptional. On the host PC, drivers are provided for PCI and USB interfacing, along with API libraries for configuration and communication with FPGA. Standard support is for C++ development, and MATLAB support is optional. For FPGA, there is special module for instantiation into the user design, which in association with API functions provides full duplex communication between the design and the host. This communication is based both on register and on DMA types of transfer.

3.2 Design Flow and Test Environment

The 802.11a modem project is based on custom rapid prototyping design flow, which is illustrated on the Fig 8.

The most important property of the design flow is linearity. This is provided by using strong FPGA chips that are not challenged with extreme performance regimes, so the timing closure iterations are avoided.

Sub-module development, as well as system integration is performed in Simulink, using the *Xilinx Blockset* for designing the models intended for implementation on FPGA. This kind of model is translated into synthesizable, target technology aware VHDL code with the *Xilinx System Generator*. This is an up-to-date tool for FPGA development, which significantly increases productivity in applications based on digital signal processing. In the Simulink, exploitation environment can be efficiently modelled. This design paradigm gives the optimum balance of the simulation effort in the process of rapid prototyping, where the deeper insight into the interaction between the design and exploitation environment is much more important than formal verification coverage. Fig.

9 shows the process of system integration and tuning, with examination of 802.11a modulator output spectrum and QAM64 constellation on the output of the demodulator.

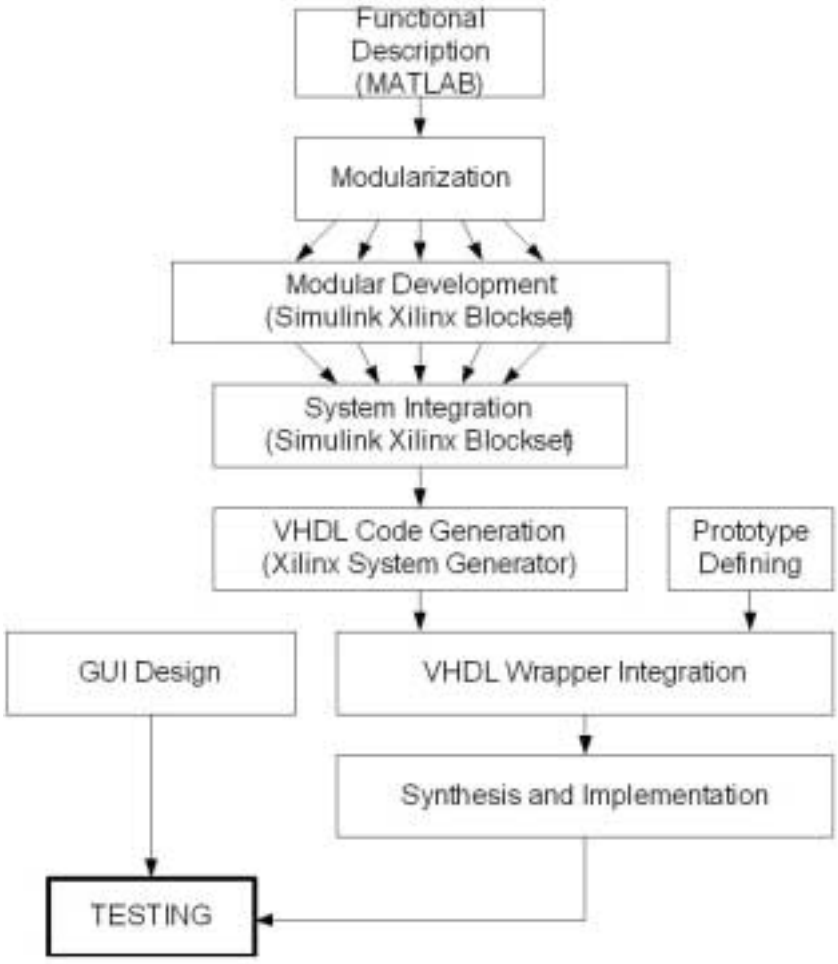

Fig. 8 - *Design flow.*

Project wrapper is a design framework written in VHDL, where 802.11a modem entity is instantiated. Project wrapper is also very specific item in the design flow shown on the Fig. 8. It provides the designer with the virtual instrumentation system, which is used for in-system probing of implemented design. List of features of the wrapper is extensive. Some of them are: data logging with event management, ADC and DAC control, data sourcing, bus controller for expanding instrumentation system with various predesigned virtual instruments, etc. Project wrapper is supported with host PC higher-level

functions written in MATLAB. These functions are based on *Nallatech MATLAB API*, and they encapsulate protocols for communication with project wrapper. On the figure 10, GUI (*Graphical User Interface*) is presented which was designed to control 802.11a FPGA prototype, and based on wrapper supporting MATLAB functions.

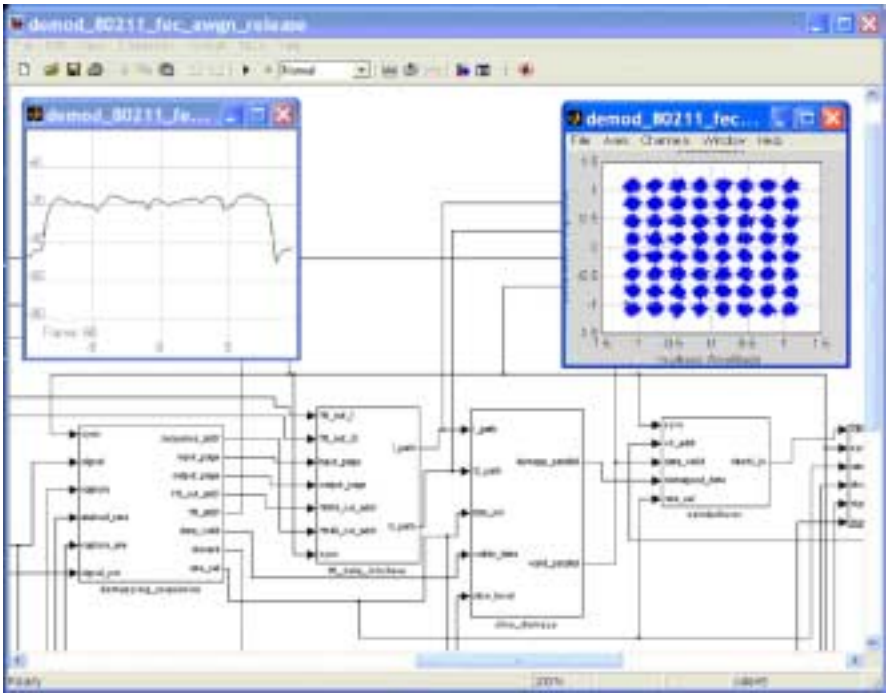

Fig. 9 - *Development in Simulink with Xilinx Blockset and System Generator.*

Some of the possibilities of GUI application are: logging the data in critical points, I/O control, constellation selection (BPSK, QPSK, QAM16 or QAM64), data source type selection, uploading coefficients for the channel emulation, the control of bit error rate measurement system, etc.

3.3 Implementation

For VHDL synthesis, *Synplicity Synplify Pro* tool is used. It is a Windows based logic synthesizer, easy to use yet very powerful. It's most specific option is *RTL View* option, which displays register transfer level logic inferred from code. For implementation flow (mapping into logic, place & route, and FPGA programming file generation), *Xilinx Foundation ISE* toolset is used. As an illustration, table 1 gives the overview of resource utilization of *Xilinx Virtex-2 XC2V2000* FPGA chip.

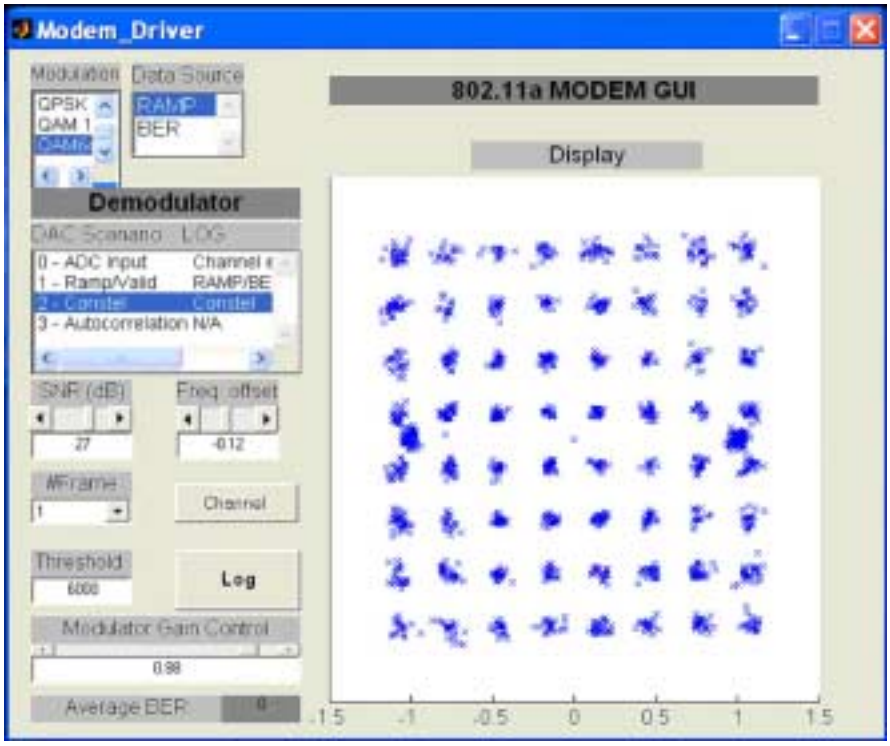

Fig. 10 - *GUI for control of 802.11a prototype.*

Table I

Resource utilization of 802.11a FPGA prototype implemented on Xilinx Virtex-2 XC2V2000.

	Logic cell (Slice) (of 10752)	Block RAM 18 kbit (of 56)	Multiplier 18x18 (of 56)
Modulator	5439 (50%)	15 (27%)	2 (4%)
Demodulator	9553 (89%)	50 (90%)	16 (29%)

4 Conclusion

In this paper, FPGA implementation of 802.11a baseband processor is presented. The selection of solutions for critical processing task is made, along with brief description of these methods. One mainstream FPGA development platform is shown, which was used for rapid prototyping in this project. Original FPGA design flow optimised for rapid prototyping is shown. In addition, resource utilization is tabulated, as a measure of design size and complexity.

The key point of this paper is to show that FPGA design paradigm gives the realistic possibility to equip and maintain high-tech laboratories in our country. Wider acceptance of FPGA could bring a new quality in EE education on one hand, and on the other it could drive more advanced project and development practice.

5 References

[1] Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, IEEE Std 802.11a-1999, 1999.

[2] R. Van Nee, R. Prasad: OFDM for Wireless Multimedia Networks, Artch House, 2001.

[3] J-J van de Beek et al: On channel estimation in OFDM systems, Proc. IEEE 45th Vehicular Technology Conference, 1995, pp. 815-819.

[4] Michael Speth et al: Optimum Receiver Design for Wireless Broadband Systems Using OFDM, IEEE Transactions on Communications, Vol. 47, 1999, pp. 1668 - 1677.

[5] www.nallatech.com