Efficient Architecture and Implementation of PHY WLAN Receiver on Reconfigurable Platforms

D.Keerthi Priya, M.Vani Devi
Department of Avionics,
Indian Institute of Space Science and Technology, Trivandrum,India
keerthidasala@gmail.com,vani@iist.ac.in

Abstract—**This paper presents the design, implementation and validation of PHYsical Layer WLAN Receiver customised for IEEE 802.11a/g standards. Since the performance requirements for receiver are fixed, the design strategy was mainly focused on the algorithms chosen for receiver in order to achieve a desired performance and also reduction in hardware cost and latency.**

I. Introduction

The Orthogonal Frequency Division Multiplexing (OFDM), which is a multicarrier modulation technique,has been adopted for several broadband communication systems; such as Wireless Local Area Networks (IEEE 802.11a/g/n) 3G and 4G cellular systems.
For packet-based transmission systems, OFDM transmitters often employ a preamble prior to data transmission.This preamble consists of ten identical short symbols (SSs) of 16 samples and two identical long symbols (LSs) of 64 samples with a guard interval (GI) of 32 samples as shown in Figure 1. It is used for synchronization and channel estimation. Figure 2 shows the block diagram of WLAN baseband transceiver. The OFDM Transmitter customised for this WLAN standards is already developed and is based on our prior work [2]. The performance is analysed taking into account the integrated transceiver system. In this work the focus is mainly on the development of the receiver. IEEE 802.11a/g introduces a latency limitation given by the Short Inter-Frame Space(SIFS) [1]. In order to account this short fall, the main parameters considered in the design are attributed to the selection of algorithms for synchronisation, channel estimation and phase tracking. The receiver design was mainly emphasised on the front end synchronizer as the performance strongly depends on it. Also our proposal is compared with other synchronization algorithms found in the literature [$3-6$].

Firstly the design of a finite precision model of each algorithm used in receiver was done followed by a fixed point analysis of the complete receiver. This step has lead to the result of quantisation of the data path of the proposed receiver, starting at the receiver input.

The fixed-point receiver was designed using Xilinx System Generator and ISE tool chain. The receiver was implemented on a prototype board connected to a logic analyzer, and its output is then verified by performing a digital loop back test under an Additive White Gaussian Noise (AWGN) channel. The main contribution of our work is the reduction in the hardware resource by which we are reducing the cost and latency.
This paper is organized as follows. Section 2 presents the 802.11a Baseband receiver architecture with the proposed synchronization algorithms and the FFT processor, the channel estimation and equalisation algorithm and the phase tracking and compensation algorithm respectively. Sections 3 describe the hardware implementation details. Section 4 deals with the description of the receiver test, the discussion of performance and a tabulated comparison with other receivers found in the literature, and the use of the hardware resources.Finally, section 5 summarizes the conclusion.

II. 802.11a Baseband Receiver Architecture

The output of the transmitter after RF demodulation is applied to the receiver unit (see Figure 2) which consists of following stages : Frame detection using RSSI(Received Signal Strength Indicator), Time Synchronisation, coarse and fine CFO estimation and correction, FFT(Fast Fourier Transform) based OFDM demodulation, channel estimation and equalisation, phase tracking and compensation, Pilot removal, Variable rate Demodulation, Deinterleaving,Viterbi decoding, Descrambling.

A. Frame detection, Coarse time and frequency synchronisation

The first step in synchronisation process is Frame detection and is done by means of energy detection based on RSSI. After that, it is necessary to estimate a time reference: $\hat{n}_{GI}$. The time estimate must be accurate to avoid ISI when the FFT windowing is done. The design rule is to accept that the synchronization is correct if deviation from the ideal initial sample n_{GI} is between 0 and -4 samples, negative offsets higher than 4 samples can cause ISI in channels with moderate to high delay spread.
For the reasons stated above, the proposed algorithm is divided into two parts: coarse and fine time synchronization. Coarse synchronization is chosen in accordance with method proposed by Schmidl and Cox [3]. Auto-correlation of the received signal with the delay of 16 samples is performed followed by averaging during 144 samples.

$$R_n = \mathbf{r_n^H r_{n+16}} \tag{1}$$

where $\mathbf{r_n} = [r_n r_{n+1} .... r_{n+143}]^T$ is a vector of 144 samples from received signal. The result of this is a single peak obtained at sample n=160 (Figure 3) where the transition between the last SS and GI is observed.

Next,the modulus of the auto-correlation output is normalized by the local mean power, $P_n = ||\mathbf{r_n}||^\mathbf{2}$, of the received

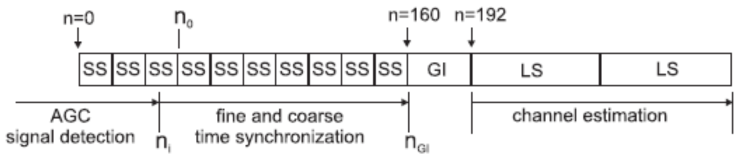


Fig. 1: Format of IEEE 802.11a preamble

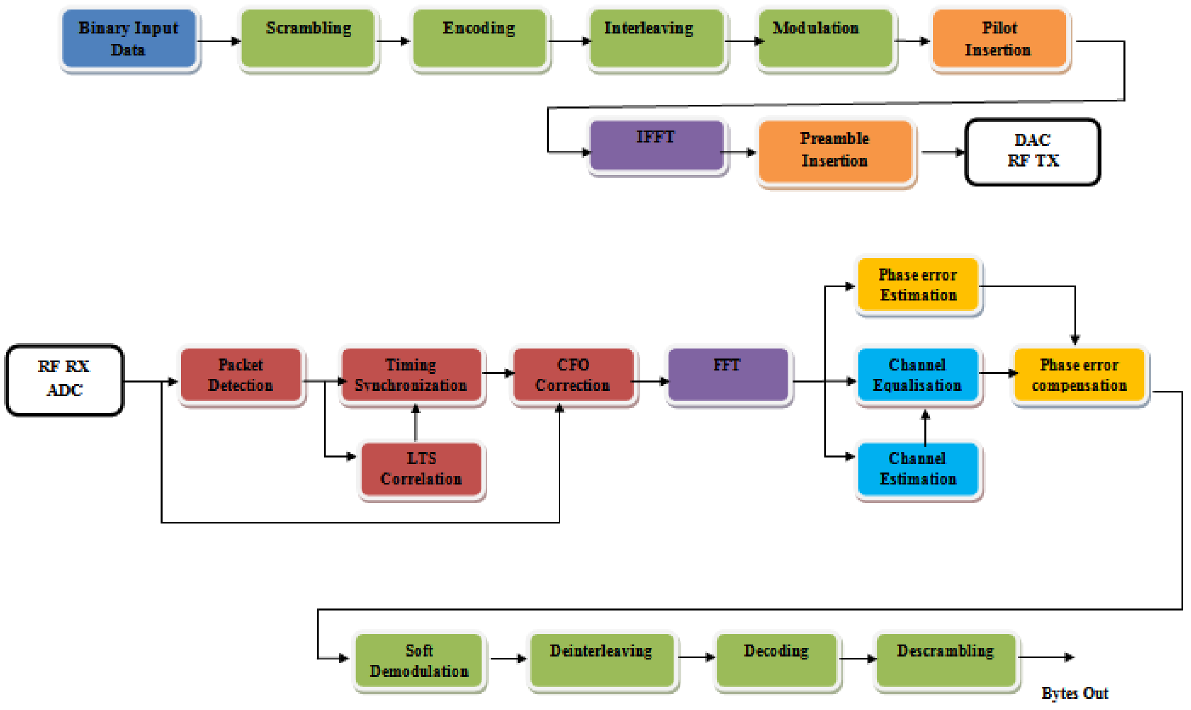


Fig. 2: Block Diagram of Baseband WLAN Transceiver

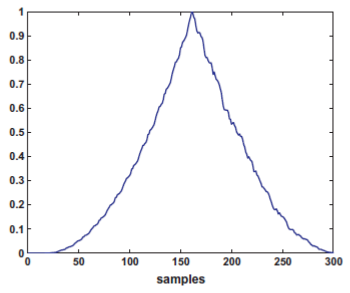


Fig. 3: Output of auto correlator $|R_n^2|$

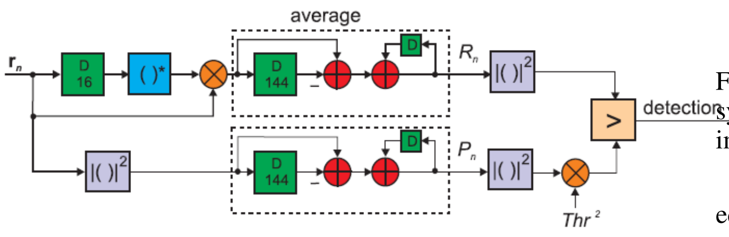


Fig. 4: Block Diagram of proposed coarse time synchronisation

signal. The timing metric is defined as [13]:

$$\bar{R}_n = \frac{|R_n|^2}{P_n^2} \quad (2)$$

Also to improve the estimation accuracy, $|R_n|^2$ is averaged during 5 samples. To find the position of the peak ($\hat{n}_{GI}$, a threshold Thr is set by searching the maximum of the averaged $|Rn|^2$ between those samples that fulfill the condition $\bar{R}_n > Thr$. To avoid the costly division operation, the condition Eqn(3) was imposed in the implemented design

$$|R_n|^2 > (P_n^2.Thr^2) \quad (3)$$

Figure 4 shows the block diagram of the proposed coarse time synchronization algorithm which can be readily mapped to implementation in VLSI.

After careful observation, a threshold of 0.4375 at SNR equal to or higher than 6 dB was fixed to guarantee the probability of not detecting the beginning of the GI is lower than 1×10^{-3}.
It was inferred that 99.9% of the frames were detected within a deviation from n_{GI} between -4 to +15 samples. In order to

account for this deviation, the use of a fine time synchronization algorithm is necessitated. The CFO($\hat{f}$) is estimated at $\hat{n}_{GI}$ as in [11]:

$$\hat{f} = \frac{\angle R_n}{2\pi 16 T_s} \quad (4)$$

where T_s is the sampling period.

The coarse synchronisation algorithm works well within the limits of maximum CFO allowed as per IEEE 802.11a/g standard: 232 kHz (73% of the subcarrier spacing); and the obtained estimates of CFO is precise enough for the highest modulation order used in the standard (64-QAM). The estimation error reported has a standard deviation of 0.35% of the subcarrier spacing for SNR higher than 20 dB. Therefore, fine frequency synchronization estimated using an autocorrelation of the LS [11], is not necessary, resulting in considerable reduction of final latency of synchronizer.
CORDIC is used in the implementation for subsequent correction of CFO from LS samples(prior to fine time synchronisation)and received OFDM symbols for which the estimated CFO($\hat{f}$) serves the purpose.

B. Fine Time synchronisation

Fine time synchronisation algorithm proposal is based on cross-correlation between the received signal and the known long training symbol (LS). For this, the LS is quantised to values -1,1 for the real and imaginary parts to avoid use of multipliers. Now the cross correlator is simply a wired complex filter similar to [12]. The length of this cross-correlation is evaluated to be 32 samples with performance similar to that of 64-sample length with no quantised LS. The reduction in length with quantised LS needs only 63 real adders in contrast to 64 complex multipliers and 63 complex adders required for 64-sample length cross-correlation with no quantised LS.

Thus, for 32-samples the cross-correlation is calculated as:

$$C_n = \mathbf{g_{32}^H r_n} \quad (5)$$

where $\mathbf{r_n} = [r_n r_{n+1} .... r_{n+31}]^T$ is a vector with 32 samples from the received signal after CFO compensation and $g_{32} = [LS_0 LS_1 .... LS_{31}]^T$ is a vector with the first 32 quantized samples from the LS. The reduction in length achieves not only the objective of minimising the hardware cost but also the latency. A latency reduction of 1.6 μs is obtained.
In ideal conditions, the cross-correlation gives a large peak at sample n = 224, that is, in the middle of the long training symbol (n_{GI+64}). Adding to this, the computational complexity is further reduced by calculating the cross-correlation only for an interval of 20 samples. This interval is the deviation given by coarse time synchronisation algorithm(between -4 and +15 samples).Therefore, the position of the peak is estimated as:

$$\hat{n}_{GI+64} = \arg max_{n1 \le n \le n2} \{|C_n^2|\} \quad (6)$$

$n1 \le n \le n2$ being the interval of 20 samples where the cross-correlation is computed ($n1 = \hat{n}_{GI} + 64 - 15$ and $n2 = \hat{n}_{GI} + 64 + 4$).

C. FFT

The boundary of each FFT is established by the synchronization blocks above. The cyclic prefix of each OFDM symbol is removed by advancing the boundary of each FFT 16 samples per transform. For each OFDM symbol, its 64 data samples are saved in the DPRAMs and, after 384 clock cycles, the FFT result is obtained. So the FFT must be calculated every 4 μs, the duration of an OFDM symbol sampled at 20 MHz, to achieve a continuous data flow. The resulting minimum clock for this is 96 MHz, but we selected 100 MHz because it is an entire multiple of the input signal frequency. For this clock, a FFT result is obtained each 3.84 μs. For the period of remaining 0.16 μs the FFT operation is disabled.

D. Channel Estimation and Equalisation

The long symbols(LS's) of the preamble are used to estimate the frequency response of the radio channel once the CFO is corrected. Here the channel estimation is performed in frequency domain i.e., after FFT operation on LS samples. The contents of the two long symbols being identical, are averaged to improve the quality of the channel estimation [11].Since FFT is a linear operation, this average can be calculated before FFT stage itself. The frequency domain response of the channel estimation ($\hat{H}$) is obtained as follows:

$$\begin{aligned} C_X &= FFT(\frac{Ls_1 + LS_2}{2}) \\ \hat{H} &= \frac{C_X}{C_L} \end{aligned} \quad (7)$$

where LS_1 and LS_2 are the first and the second received LS, respectively, and C_L is the transmitted LS in the frequency domain.

The channel distortion is equalised using Zero Forcing Solution which is simple compared to other equalisation methods. For this the received OFDM symbols are multiplied by the inverse of the channel estimation. To circumvent the division operation by a complex number (C_X) the inverse is calculated otherwise as:

$$\frac{1}{\hat{H}} = \frac{C_L}{C_X} = \frac{1}{C_X.C_X^*}.C_X^*.C_L = \frac{1}{|C_X|^2}.C_X^*.C_L \quad (8)$$

The implementation for the channel estimation and equalisation is as shown in Figure 6.

E. Phase Tracking and Compensation

An observation shows that there is rotation in the phase which is constant for the subcarriers and is incremented from one OFDM symbol to another. This rotation further causes a rotation in the constellation and it is seemingly difficult to perform accurate demodulation after receiving few OFDM symbols. This is mainly due the effect of residual CFO. To account for this, phase tracking i.e., estimation of rotation followed by compensation for each OFDM symbol is performed. This scheme makes use of the four pilot subcarriers embedded in the OFDM symbols. The phase rotation is detected by comparing the received pilot subcarriers (R_{nk}) against the known pilot subcarriers (P_{nk}) in the frequency domain. The

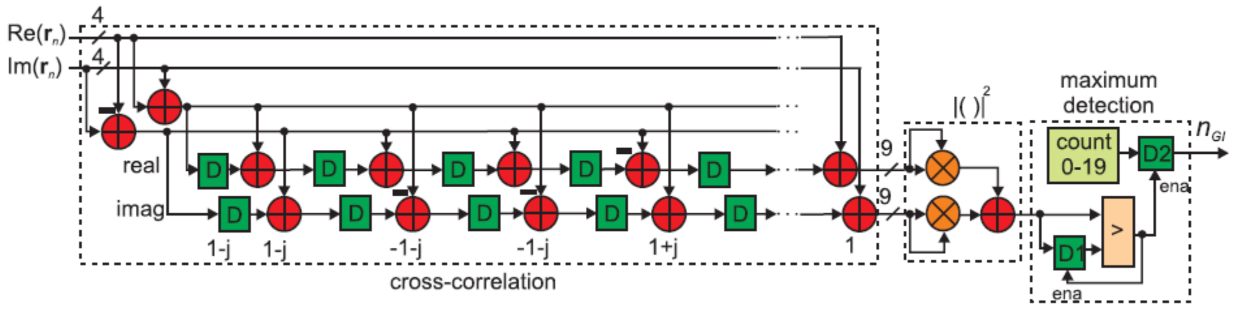


Fig. 5: Proposed scheme for fine time synchronisation

estimated channel frequency response $\hat{H}_k$ is used to estimate the phase estimate $\hat{\phi}_n$ given by

$$\hat{\phi}_n = \angle[\sum_{k=1}^{4} R_{nk}(\hat{H}_k.P_{nk})^*] \tag{9}$$

For each OFDM symbol, CORDIC is reused to estimate the phase as well as compensation of the equalised data subcarriers as shown in Figure 7.

F. Symbol Demodulation

Demodulates the equalized constellation symbols and obtains the log likelihood ratios (LLR) of the received bits which will be used by the soft decision Viterbi decoder. For BPSK and QPSK modulation types, the LLR of each bit can be easily calculated by evaluating only the in-phase or quadrature components of the constellation symbols. However, the calculation of LLR is more difficult for the 16-QAM and 64-QAM modulation types, because each component of the constellation carries two and three bits of information, respectively.

$$\begin{aligned}
BPSK: LLR(b_0) &= [y_I[n] \times 32]\\
QPSK: LLR(b_0) &= [y_I[n] \times \sqrt{2} \times 32]\\
16-QAM: LLR(b_0) &= [y_I[n] \times \sqrt{10} \div 3 \times 40]\\
LLR(b_1) &= [(2 - |y_I[n] \times \sqrt{10}|) \div 3 \times 40]\\
64-QAM: LLR(b_0) &= [y_I[n] \times \sqrt{42} \div 7 \times 48]\\
LLR(b_1) &= [(4 - |y_I[n] \times \sqrt{42}|) \div 7 \times 48]\\
LLR(b_2) &= [(2 - |4 - |y_I[n] \times \sqrt{42}||) \div 7 \times 48]
\end{aligned} \tag{10}$$

G. De-Interleaver

Like the transmitter the deinterleaving at the receiver is also governed by two permutation. At the receiver let j denote the index of the received signal and i denotes the index after the first interleaving and k denotes the index after the second permutation.

The first permutation is defined by the rule

$$i = s \times Floor(\frac{j}{s}) + (j + Floor(16\frac{j}{N_{CBPS}})) \text{ mod s j} = 0, 1, N_{CBPS} - 1 \tag{11}$$

where $s = max(\frac{N_{BPSC}}{2}, 1)$

The second permutation is defined by the rule

$$k = 16i - (N_{CBPS} - 1)Floor(16 \times i/N_{CBPS}), \text{ i} = 0, 1, N_{CBPS} - 1 \tag{12}$$

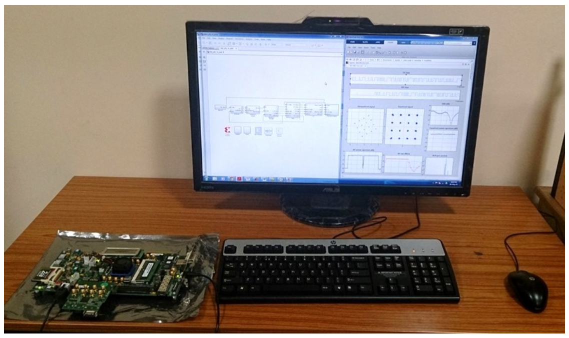
H. Viterbi Decoder

The Viterbi Decoder uses a trellis based decoding. The convolution encoder at the transmitter uses industrial generator polynomial $[133\ 171]_8$ to produce a 1/2 rate convolutional code with constraint length 7. The receiver employs a trellis based maximum likelihood Viterbi decoder which decodes the input bits to obtain the information bits. The trellis length is chosen to be 5 times the constraint length. For this implementation, an open source verilog implementation of the trellis based Viterbi decoder [13] has been used. It has been incorporated into the transceiver design as a black box along with proper interface ports to connect to other adjoining blocks.

I. DeScrambler

The final stage of the OFDM receiver is frame synchronous descrambler that reorders the randomised received signal according to the same logic as set in the scrambler in transmitter section. The initial value of the descrambler is set to the service field value extracted from the PLCP header.

III. Hardware Implementation



Three different input signals were used to test the transceiver system. (1) QPSK (18 Mbits/s), SNR = 20 dB and CFO between transmitter and receiver of 150 kHz; (2) 16-QAM (36 Mbits/s), SNR = 25 dB and CFO = 150 kHz; and (3) 64- QAM (54 Mbits/s), SNR = 30 dB and CFO = 150 kHz. The performance was checked for each operation mode

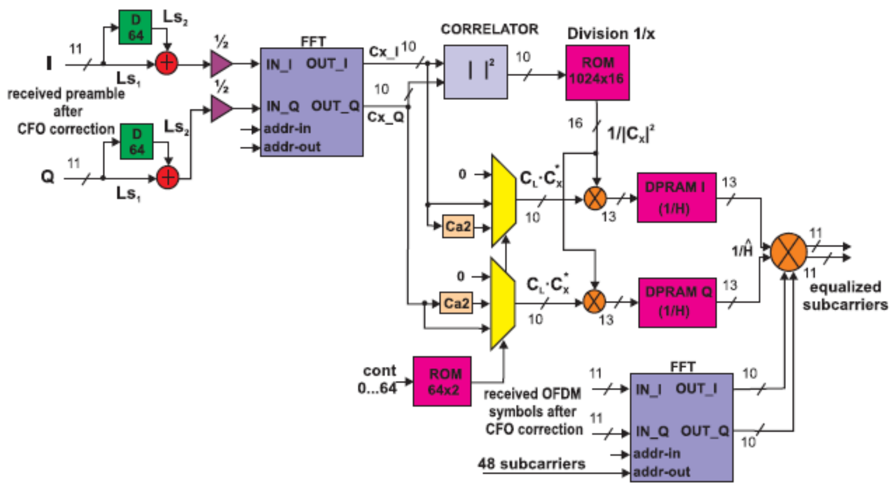


Fig. 6: Implementation of channel estimation and compensation

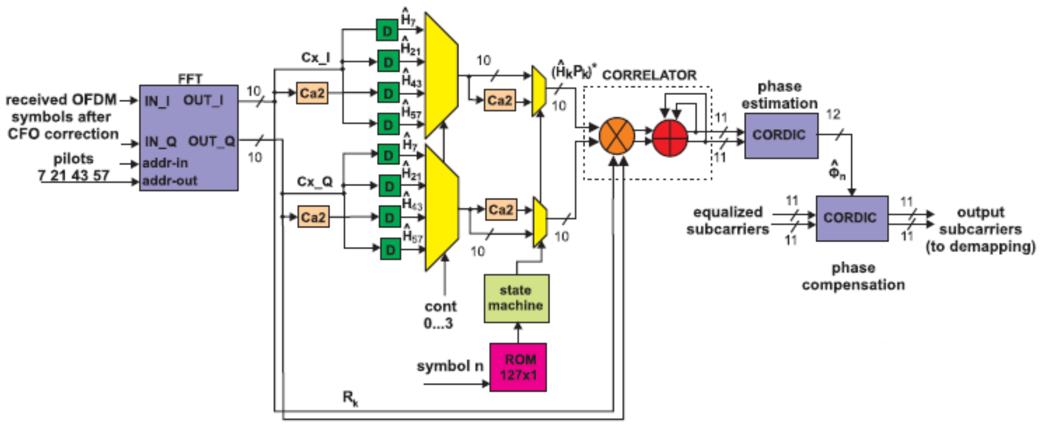


Fig. 7: Implementation of phase tracking and compensation

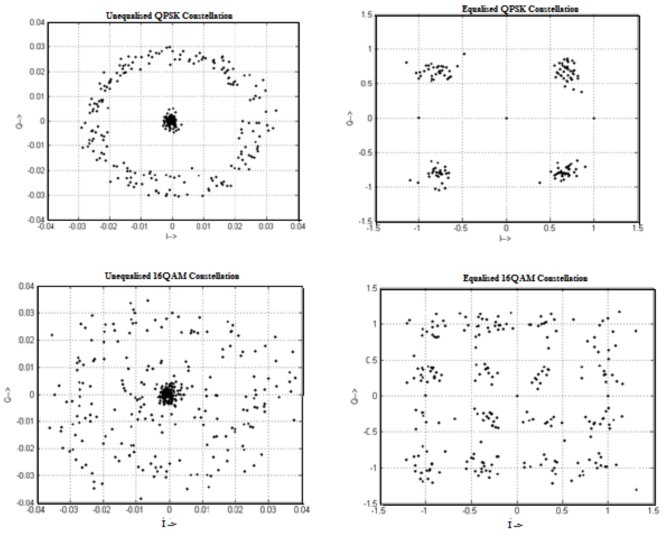
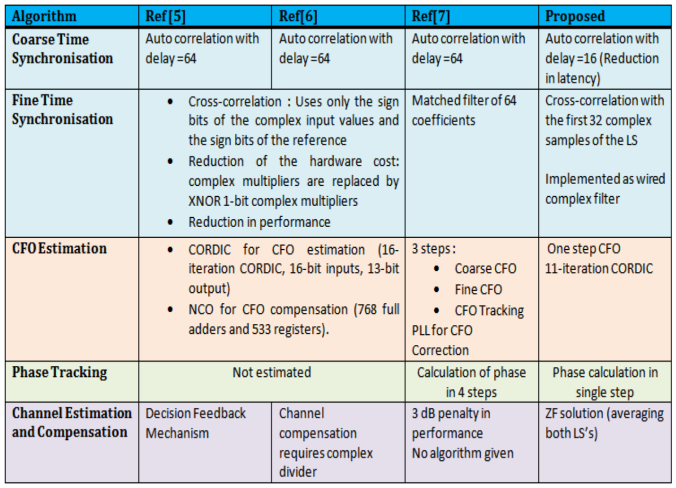


Fig. 8: Run time output constellation capture for QPSK and 16 QAM

with error correction stage involved and the SNR was noted which guaranteed a BER lower than 10^{-5}. Firstly, the designed receiver was implemented on a Virtex-VI FPGA Board, which was connected to a logic analyzer . The output subcarriers were compared to the fixed point model of receiver under similar test conditions. As seen in Figure 8 we obtained the desired constellation before and after channel equalisation for each input signal.

IV. Performance Results and Receiver Structure Comparison

The performance results of the proposed receiver including all the stages were obtained under simulation conditions which were: AWGN channel model, 64-QAM (54 Mbits/s), error correction (Viterbi with CSI [10]), 1000 frames (each frame was composed by 50 data packets of 1000 bytes).
A comparison in terms of algorithm implementation,cost and performance was made between our receiver and other solution proposed in the literature [6-9].Due to the fact that different technologies and implementation strategies were utilized it is extremely difficult to make a fair comparison with these solutions. Figure 9 shows the comparison of all the algorithms



Algorithm	Ref[5]	Ref[6]	Ref[7]	Proposed
Coarse Time Synchronisation	Auto correlation with delay =64	Auto correlation with delay =64	Auto correlation with delay =64	Auto correlation with delay =16 (Reduction in latency)
Fine Time Synchronisation	• Cross-correlation : Uses only the sign bits of the complex input values and the sign bits of the reference • Reduction of the hardware cost: complex multipliers are replaced by XNOR 1-bit complex multipliers • Reduction in performance		Matched filter of 64 coefficients	Cross-correlation with the first 32 complex samples of the LS Implemented as wired complex filter
CFO Estimation	• CORDIC for CFO estimation (16-iteration CORDIC, 16-bit inputs, 13-bit output) • NCO for CFO compensation (768 full adders and 533 registers).		3 steps : • Coarse CFO • Fine CFO • CFO Tracking PLL for CFO Correction	One step CFO 11-iteration CORDIC
Phase Tracking	Not estimated		Calculation of phase in 4 steps	Phase calculation in single step
Channel Estimation and Compensation	Decision Feedback Mechanism	Channel compensation requires complex divider	3 dB penalty in performance No algorithm given	ZF solution (averaging both LS's)

Fig. 9: Table showing the algorithm comparison of proposed and existing receiver solution

proposed in literature relevant to our present proposal.Taking into account the simplicity of implementation,algorithm complexity and the latency factor, it is observed that our receiver is architecturally efficient and requires less hardware resources than the receiver proposed in [6]. Finally in contrast our design includes a complete fixed point analysis and PER performance of the implemented receiver.

Table 1 summarizes the SNR at a 10% PER required by 802.11a standard and the studied receivers, for AWGN channel and different data rates.As can be seen, in comparison to the 802.11a standard ,our proposal outperforms the receivers implemented previously and a smaller SNR is needed at every data rate.

Also the results highlight that the transceiver prototype

Rate (Mbps)	802.11a/g required SNR (dB)	Achieved SNR (dB)	Ref [7] (dB)	Ref[8] (dB)	Ref [9] (dB)
9(BPSK)	10.7	3.9	5.8	5.8	9.7
18(QPSK)	14.7	8.1	9.9	9.5	12.8
36(16-QAM)	21.7	14.6	15.9	14.9	20.5
54(64-QAM)	26.7	19.6	21.7	20.6	26.1

TABLE I: Performance comparison at 10% PER

implementation as shown in Table II used 7022 Slices, 20283 LUTs, 56 BRAM memory blocks, 21 DSP48E1s, 350 IOBs of the Xilinx XCV6LX240T FPGA chip corresponding to 12% of the total resources and has an improvement over [10].

	Module	Slices (37680)	LUTs (150720)	BRAMs (832)	DSP48E1s (768)	IOBs (600)
Proposed	Transmitter	1065	2208	17	14	350
	Receiver	5899	18053	16	82	398
	Transceiver	7022	20283	21	56	373
Ref[10]	Transmitter	2796	5594	32	15	– –
	Receiver	7188	16589	12	48	– –
	Transceiver	10333	22704	57	64	– –

TABLE II: Hardware Utilization of Complete Radio system

V. Conclusion

In this work, the implementation of the OFDM based WLAN receiver on FPGA has been presented.Emphasis is given to the synchronizer design because the performance of the receiver strongly depends on it. The proposed synchronization algorithm outperforms the ones in [3 - 6].Additionally, our synchronizer has low hardware cost and low latency. In any case, the complete receiver achieves the desired performance: it outperforms the receivers implemented previously [7 - 9] and needs smaller SNR at every data rate than the one required by the 802.11a standard. The designed receiver is implemented on a prototype board in order to verify it: Firstly the output subcarriers are checked with those obtained by simulation and next the output constellations are observed for different modulation schemes.

In conclusion, the proposed implementation of an OFDM-based WLAN receiver achieves excellent performance with low hardware cost and low latency (the first output subcarrier is obtained 7.2 μs after the first data sample of the first OFDM data symbol is received).

Acknowledgements

We express our sincere thanks to *"Agilent Technologies"* for providing all the hardware necessary to carry out our simulation and tests.

References

[1] IEEE Standard for Information technology-Telecommunications and information exchange between systems Local and metropolitan area networks- Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 2012

[2] D.Keerthi Priya, M.Vani Devi, *"Rapid Prototyping Framework for PHY WLAN OFDM Transmitter on Reconfigurable Platforms"*,International Journal of Multidisciplinary Education Research, Volume-3, Issue-3(10),2014.

[3] T. Schmidl, D. Cox, *"Robust frequency and timing synchronization for OFDM"*, IEEE Transactions on Communications 45 (12) (1997).

[4] Yong Soo Cho,Jaekwon Kim,Won Young Yang,Chung G. Kang, *"MIMO OFDM Wireless Communications with MATLAB"*,IEEE Press,JohnWiley and Sons Pvt Ltd.

[5] A. Troya, K. Maharatna, M. Krstic, E. Grass, U. Jagdhold, R. Kraemer, *"Efficient inner receiver design for OFDM-based WLAN systems: algorithm and architecture"*, IEEE Transactions on Wireless Communications 6 (4) (2007) 13741385.

[6] A. Troya, K. Maharatna, M. Krstic, E. Grass, U. Jagdhold, R. Kraemer, *"Low-power VLSI implementation of the inner receiver for OFDM-based WLAN systems"*, IEEE Transactions on Circuits and Systems I 55 (2) (2008) 672686.

[7] Wei-Hsiang Tseng, Ching-Chi Chang, Chorng-Kuang Wang, *"Digital VLSI OFDM Transceiver Architecture for Wireless SoC Design"*, ISCAS, 2005.

[8] J. Thomson et al., *"An Integrated 802.11a Baseband and MAC Processor"*, Dig. Tech. Papers IEEE ISSCC 2002, vol. 1, February 2002, pp. 126127.

[9] T. Fujisawa et al., *"A single-Chip 802.11a MAC/PHY with a 32-b RISC processor"*, IEEE Journal of Solid State Circuits 38 (11) (2003) 20012009.

[10] Aveek Dutta *"CODIPHY- Composing On-Demand Intelligent PHYsical Layers"* Doctoral Thesis - 2013

[11] J. Heiskala, J. Terry, *"OFDM Wireless LANs: A Theoretical and Practical Guide"*, SAMS Publishing, 2001.

[12] M.J. Canet, I. Wassel, V. Almenar, J. Valls, *"Performance evaluation of fine time synchronizer for WLANs"*, in: Proceedings of 13th European Conference on Signal Processing (EUSIPCO 2005), September, 2005.

[13] Perl Script for Viterbi Decoder -. [Online]. Available: http://viterbi-gen.sourceforge.net/