Performance Evaluation of Time and Frequency Domain Equalizers

Anis Souari
Department of Electrical and Computer Engineering
Concordia University
Montreal, Quebec, Canada
asouari@ece.concordia.ca

Mohamed L. Ammari
Innov'Com Laboratory
École Nationale d'Ingénieurs de Sousse
Sousse, Tunisia
mohamed.ammari@eniso.rnu.tn

Amjad Gawanmeh
Department of Electrical and Computer Engineering
Khalifa University
Abu Dhabi, UAE
amjad.gawanmeh@kustar.ac.ae

Sofiène Tahar
Department of Electrical and Computer Engineering
Concordia University
Montreal, Quebec, Canada
tahar@ece.concordia.ca

***Abstract*—This paper presents performance evaluation of two implementations of an equalizer: a time domain equalizer (TDE) based on the Least Mean Squares (LMS) algorithm and a frequency domain equalizer (FDE) based on the Fast LMS algorithm. The comparison between the two algorithms is based on the computational complexity and resources. The computational complexity of the two algorithms is analyzed by simulation of the TDE and FDE at at two levels of abstraction: the design specification based on floating point arithmetics using Simulink, and the design implementation based on fixed point arithmetics using Xilinx's System Generator tool. The models are used to measure both floating-point and fixed-point signal-to-noise ratio (SNR) errors based on the two algorithms and provide error estimation for the design specification and design implementation. We analyze the resources used in the implementation of the two algorithms by providing FPGA implementations in System Generator. Our analysis shows that the FDE is more efficient in terms of computational complexity and resources.**

I. Introduction and Motivation

The channel equalization is an application of adaptive filtering that can eliminate the inter-symbol interference caused by the multipath phenomena. To decrease the filtering complexity, the equalizer can be implemented in the frequency domain using the Fast Fourier Transform (FFT) or the Inverse FFT (IFFT), where time convolution is replaced by frequency multiplication. This method offers low complexity growth in comparison with the time domain method.

Different algorithms have been proposed for the design and implementation of equalizers, in both time and frequency domains. The equalizer can be implemented using the Least Mean Squares (LMS) algorithm in time domain, or the Fast LMS (FLMS) algorithm in the frequency domain [1]. The performance of these algorithms has not been tested in terms of error analysis at several levels of abstraction including different number domains. In addition, the efficiency of the different methods have not been considered in terms of hardware implementations in FPGA. In this work, we build on our previous work to provide a method for the performance analysis of time domain equalizer (TDE) and frequency domain equalizer (FDE) based on error estimation and FPGA implementation. In our previous works [2], a design for verification methodology was proposed based on theorem proving and simulation techniques in order to provide the error analysis for an implementation of the FDE based on the FLMS algorithm.

In this work, the performance of the equalizer is evaluated based on two different algorithms: LMS algorithm in time domain and the FLMS algorithm in frequency domain. The error estimation and resources utilization metrics are used for the comparison. In the first, we conduct simulations for design models based on the two algorithms in order to measure floating-point and fixed-point SNR errors. The SNR floating-point error estimation is performed by comparing the SNR error of the FDE and the SNR error of the TDE models that were both implemented in Simulink [3]. On the other hand, the fixed-point error estimation is performed by comparing the SNR error of the FDE and the SNR error of the TDE models that were both implemented using System Generator for DSP [4]. We then provide an FPGA based implementation for the LMS algorithm in the time domain for the FLMS algorithm in the frequency domain. Finally, we compare between the time domain and frequency domain based designs in terms of two metrics: floating-point and fixed-point SNR error estimation (Simulink and System Generator models) and resources used in FPGA (DSP model). We show that the equalizer is more efficient when implemented in the frequency domain, hence it was further verified using formal error analysis.

The rest of the paper is organized as follows: Section II overviews the performance analysis methodology. Section III presents the error estimation performance analysis in Simulink. Section IV describes the FPGA implementation of time domain and frequency domain equalizers. Section V discusses the comparison between both equalizers in terms of performance and resources and discusses contributions compared to related work. Finally, Section VI concludes the paper.

II. Performance Analysis Methodology

Figure 1 shows the first part of the design comparison methodology. The equalizer is designed using the LMS algorithm (and its variation Block LMS) in the time domain. On the other hand, the FLMS is used to design the equalizer in the frequency domain. From these two algorithms, design specifications are obtained based on floating-point and fixed-point arithmetic. Then, we modeled each design in a Simulink model using floating-point arithmetic. Simulink is used in order to estimate the SNR for the design in time domain and the SNR for the design in the frequency domain, where both errors were estimated based on the floating-point arithmetic model. Using these two SNR values, SNR error estimation is obtained for the floating-point level design.

Next, we build a System Generator design implementation model for the TDE with the LMS algorithm and the FDE with the FLMS algorithm. Both of these design implementations are obtained using fixed-point arithmetic. Similar to the floating-point level design, System Generator is used to provide a DSP implementation in order to estimate the SNR for the design in both time and frequency domains, where both errors were estimated based on the fixed-point arithmetic model. Based on these two SNR values, SNR error estimation is obtained for the fixed-point level design. For both Simulink and System Generator models, at each level (floating-point and fixed-point), we performed SNR error estimation which is obtained in every step of the simulation for a number of iterations. The comparison shows that the FDE outperforms the TDE.

Finally, we provide an FPGA based implementation for the LMS algorithm in the time domain and another FPGA based implementation for the FLMS algorithm. We compare between the time domain and frequency domain based designs using the floating-point and fixed-point SNR error estimation and the computational complexity and resources used in FPGA. We show that the equalizer is more efficient when implemented in the frequency domain. The importance of the comparison between the time domain and frequency domain equalizers consists in the fact that considering better performance design will help in avoiding several design problems at later stages, and makes it easy to verify the reliability of the design.

III. Error Estimation based Performance Analysis in Simulink

The performance analysis of the equalizer specification is obtained by estimating the error generated in every step. The accuracy of the verification process was affected thoughtfully by the method used in the framework to model numbers, be it floating-point or fixed-point. This section provides error estimation for the two algorithms by conducting simulations at two levels of abstraction: floating point design specification models, and fixed point design implementation models. We then compare the performance of the two algorithms at each level. Simulations are carried out using the 16-QAM modulation in the Proakis channel-A [5] with additive white Gaussian noise (AWGN) [6].

A. *Time Domain Equalizer*

The TDE designed with the LMS algorithm was simulated using Matlab. We simulate two types of time domain equalizers, the first is based on the LMS algorithm, and the second is based on the BLMS algorithm. The difference between the two equalizers is related to updating the coefficients, which is done after each sample for the first and after an input block of

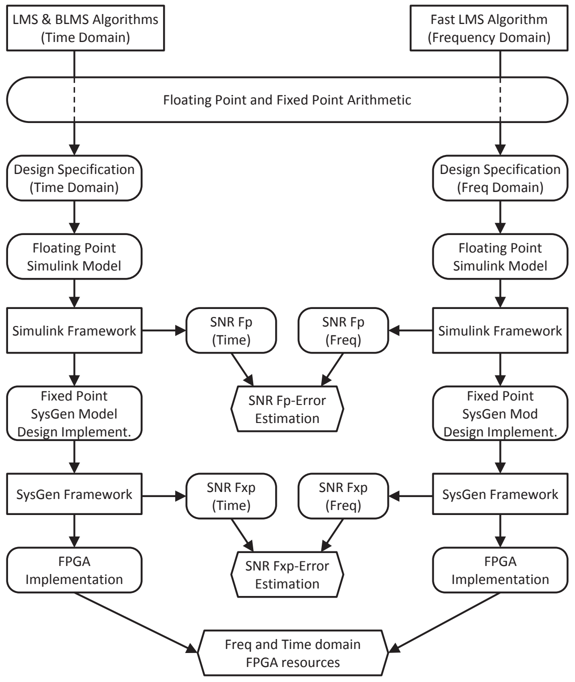
LMS & BLMS Algorithms (Time Domain)
Fast LMS Algorithm (Frequency Domain)
Floating Point and Fixed Point Arithmetic
Design Specification (Time Domain)
Design Specification (Freq Domain)
Floating Point Simulink Model
Floating Point Simulink Model
Simulink Framework
SNR Fp (Time)
SNR Fp (Freq)
Simulink Framework
Fixed Point SysGen Model Design Implement.
SNR Fp-Error Estimation
Fixed Point SysGen Mod Design Implement.
SysGen Framework
SNR Fxp (Time)
SNR Fxp (Freq)
SysGen Framework
FPGA Implementation
SNR Fxp-Error Estimation
FPGA Implementation
Freq and Time domain FPGA resources

Fig. 1. Performance Analysis of Time and Frequency Domain Equalizers

samples for the second, resepctively. The following simulation parameters for channel characteristics are used throughout all our simulations: $SNR = 40dB$, transmitted sequence length $N = 10000$, and adaptive filtering step-size $\mu = 0.002$.

Figure 2 gives the constellations of the received signal and the equalized signal. We can see that the 16-QAM constellation of the received signal is spread, while the sixteen symbols of the TDE are completely separated. Error estimation for this operation is shown in Figure 3. The above curve starts with high error since the equalizer is still updating its coefficients. After around 2000 symbols, the error reaches a state value that is equal to 25 dB, where the SNR is equal to 40 dB. Using the LMS algorithm as an adaptive algorithm for the equalizer gives an error rate equal to 1.91%. This means that on average we expect to loose 191 symbols when sending 10000 symbols. We also simulate a BLMS based equalizer using Matlab with the same LMS equalizer simulation parameters.

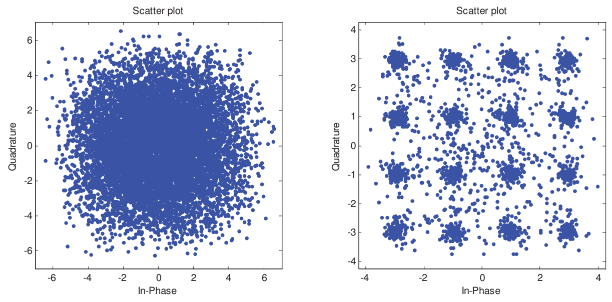

Fig. 2. (a) Signal Constellation before LMS-based Equalization, (b) Signal Constellation after LMS-based Equalization (Time Domain)

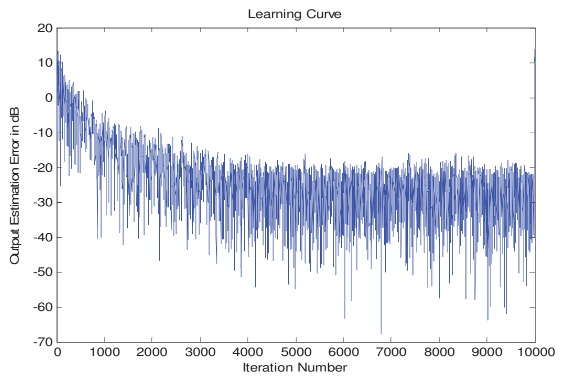

Fig. 3. LMS Equalizer Error Estimation in Time Domain

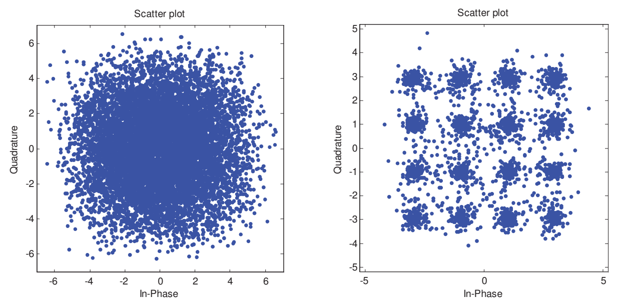

Fig. 4. (a) Signal Constellation before BLMS-based Equalization, (b) Signal Constellation after BLMS-based Equalization

The Block LMS (BLMS) equalizer is a time domain equalizer that succeeds to eliminate the Inter Symbol Interference (ISI) and gives almost the same performance as the LMS equalizer since the error rate given by the BLMS equalizer is 2.03%. The error behavior for the BLMS equalizer is similar to that of the LMS, as it becomes stable at around -25 dB. However, the BLMS algorithm converges slower than the LMS algorithm since it requires 3000 symbols to converge. The constellations of the equalizer input and output signal are shown in Figure 4 and error estimation curve for the BLMS is shown in Figure 5.

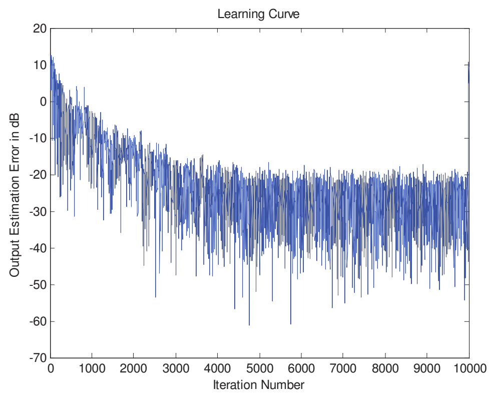

Fig. 5. BLMS Equalizer Error Estimation in the Time Domain

The floating-point model of the TDE is implemented using a transmission chain of the time domain LMS-based equalizer in Simulink. The simulated system uses 4-tap TDE based on the LMS algorithm. The Simulink model for the LMS algorithm block diagram is well described in [7].

In addition to the desired signal, the equalizer receives a noisy signal coming from the channel. The equalizer produces an output signal composed of of the equalized signal and the equalization error estimation. In order to estimate this error, simulation is conducted in Simulink where design parameters were set as $SNR = 40dB$ and $\mu = 0.002$. The efficiency of the equalizer is shown in Figure 6 which shows the constellation of the equalizer output signal. The error estimation curve, given in Figure 7, shows that the 4-tap TDE converges into a steady value of -35 dB after around 800 symbols.

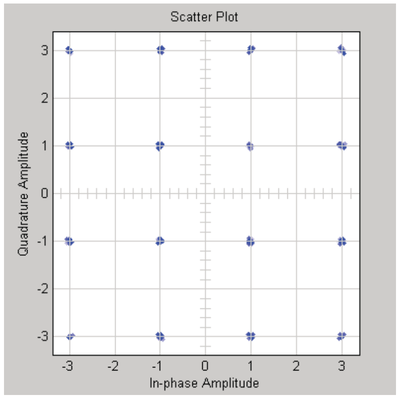

Fig. 6. Signal Constellation after Time Domain Equalization in Simulink

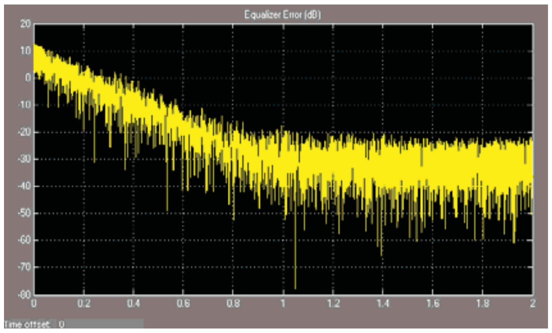

Fig. 7. Error Estimation in Time Domain

B. *Frequency Domain Equalizer*

The implementation of the FDE is based on an adaptive FLMS algorithm. We have used the Matlab/Simulink environment in order to test its performances. Design parameters used for the FDE simulation are similar to those used for the TDE simulation. The constellations of the received and equalized signals are shown in Figure 8. We can easily see that the FDE eliminates the ISI. Figure 9 illustrates the equalizer error estimation for the FLMS in the frequency domain. It is shown that the FLMS equalizer error estimation curve converges within only 200 symbols to reach the -40 db floor for an SNR of 40 dB. This implies that the FDE is more efficient than the TDE.

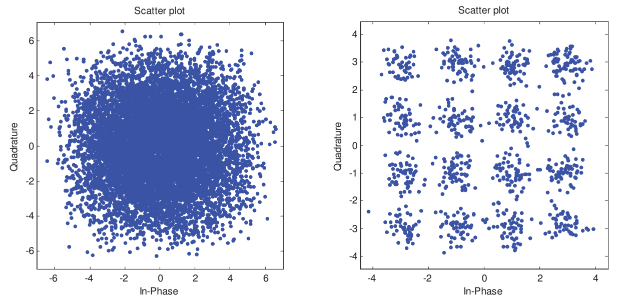

Fig. 8. (a) Signal Constellation before FLMS-based Equalization, (b) Signal Constellation after FLMS-based Equalization (Frequency Domain)

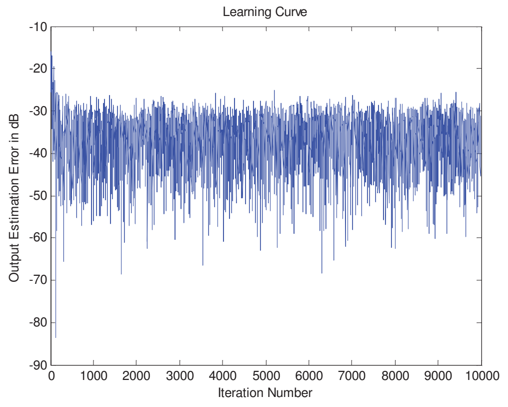

Fig. 9. FLMS Equalizer Error Estimation in the Frequency Domain

The architecture of the FLMS-based equalizer with 4 taps is modeled in the frequency domain in the Simulink environment. For this architecture, three FFT blocks and two IFFT blocks are used to allow the alternation between the time and the frequency domain. The Simulink model for the FLMS algorithm block diagram can be found in [7]. Figure 10 shows the constellation of the equalizer output signal. The constellation shows no noise which implies that the equalizer is working perfectly. Similar to the FDE, the 4-tap FDE converges after around 200 symbols to a steady value of -40 dB. The error estimation curve for the 4-tap equalizer is shown in Figure 11.

IV. Performance Analysis of FPGA Implementations

The FDE and TDE are implemented using FPGA Xilinx block set [8], then simulation is conducted for the FPGA model in order to obtain their error estimation.

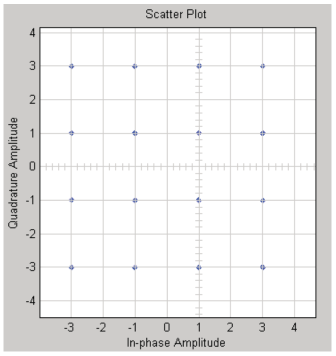

Fig. 10. Signal Constellation after Frequency Domain Equalization in Simulink

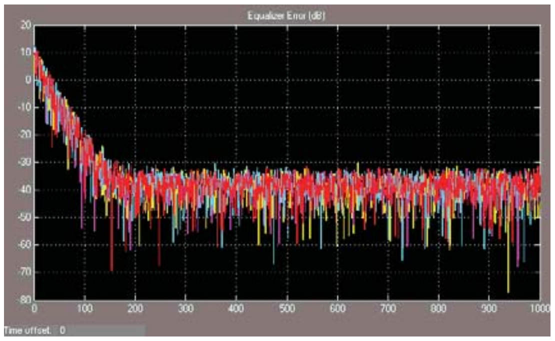

Fig. 11. Error Estimation in the Frequency Domain

A. *Time Domain Equalizer*

To get a synthesizable version of the TDE, it must be described and simulated using the specific Xilinx block set in the Simulink environment. The signal constellations and the error estimation curve are given in Figures 12 and 13, respectively. Comparing the results given by the System Generator description and the standard Simulink blocks description of the 4-tap TDE, we can deduce that the Simulink design gives better results than the System Generator one. In fact, in Simulink, floating-point is used to describe the symbols, while fixed-point is used in System Generator. Hence, the Simulink design gives more accurate results than the System Generator design. The FPGA implementation is automatically obtained by the System Generator for DSP tool which uses Xilinx ISE 11.1 [8]. The design has been implemented in a Spartan 3 FPGA board with one million gates. Table I shows the FPGA logic blocks consumption by the design including flip flips, Lock-Up-Tables (LUTs), combinational logics, IO Buffers (IOB), shift registers, route-through connectors, etc. It is obvious that the 4-tap LMS equalizer consumes almost 80% of the FPGA resources.

B. *Frequency Domain Equalizer*

The FDE is described using the Xilinx block set in the Simulink environment. We tried to implement a 4-tap FDE on the one million gate Spartan 3 FPGA board but the design was

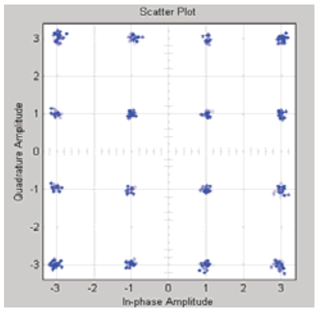

Fig. 12. Signal Constellation after Time Domain Equalization in System Generator

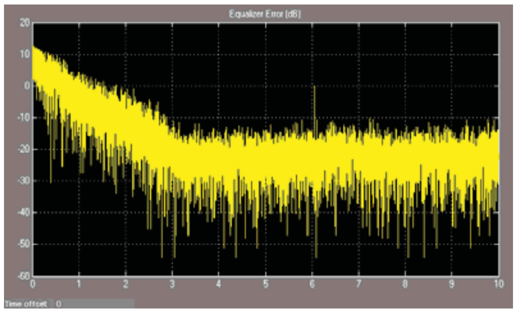

Fig. 13. Error Estimation Curve (Time Domain)

too big and therefore we implemented a 2-tap equalizer. The design contains elementary blocks from the Xilinx block set which are multipliers and adders and also some sub-systems performing complex multiplication, FFT and IFFT. The signal constellations and the error estimation curve given in Figures 14 and 15, respectively, show that the ISI are canceled. The FPGA implementation was done using the same one million gate Spartan 3 FPGA board. Table II shows the statistics of the used logic to implement the FDE design.

V. Discussion

In the LMS algorithm, getting an output sample requires N complex multiplications which mean that generating an N

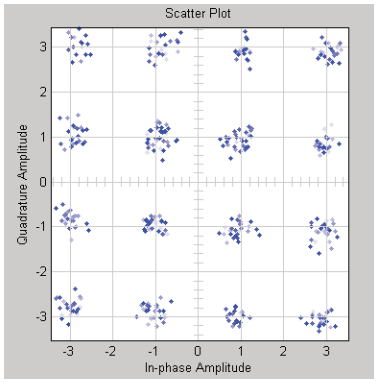

Fig. 14. Signal Constellation after Frequency Domain Equalization in System Generator

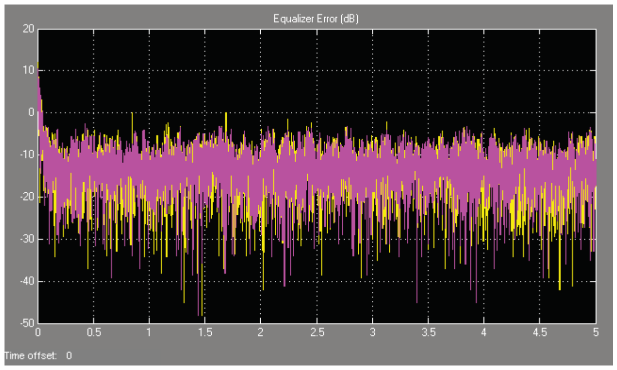

Fig. 15. Error Estimation Curve (Frequency Domain)

sample output block requires N^2 complex multiplications. In addition, updating the filter coefficients requires N^2 complex multiplications. Therefore, the LMS algorithm complexity is equal to $2N^2$ complex multiplications, which is equivalent to $8N^2$ real multiplications. On the other hand, the complexity of the FLMS algorithm is the result of the FFT and IFFT blocks. Since the computational complexity of an $2N$ FFT block is equal to $(N/2)\log_2(2N)$ complex multiplications, the 5 FFT and IFFT blocks require $(5N/2)\log_2(2N)$ complex multipli-

TABLE I
Time Domain Equalizer Synthesis Report

Logic Utilized	**Available**	**Used**	**Utilization**
Number of slice flip flops	144	15360	1%
Number of 4-input LUTs	12230	15360	79%
Number of occupied slices	6278	7680	81%
Slices containing related logic	6278	6278	100%
Slices containing non-related logic	0	6278	0%
Total Number of 4-input LUTs	12240	15360	79%
Number of used logic	12086		
Number of used as route-through	10		
Number of used as shift-register	144		
Number of bonded IOBs	233	391	59%
Number of bonded BUFGMUXs	1	8	12%
Average fanout of non-clock nets	2.93		

TABLE II
Frequency domain equalizer synthesis report

Logic Utilized	**Available**	**Used**	**Utilization**
Number of slice flip flops	56	15360	1%
Number of 4-input LUTs	9157	15360	59%
Number of occupied slices	4818	7680	62%
Slices containing related logic	4818	4818	100%
Slices containing non-related logic	0	4818	0%
Total Number of 4-input LUTs	9173	15360	59%
Number of used logic	9157		
Number of used as route-through	16		
Number of bonded IOBs	201	391	51%
Number of bonded BUFGMUXs	1	8	12%
Average fanout of non-clock nets	2.49		

cations which is equivalent to $10N\log_2(2N)$ real multiplications. The FLMS algorithm requires also two $2N$ complex vector products adding $16N$ real multiplication. Finally, the cost of the FLMS algorithm is equal to $10N\log_2(2N)+16N$. The number of the real multiplications required by the two algorithms for N symbol blocks is given in Table III below where N is equal to $2, 4, 8, 16$ and 32.

TABLE III
EQUALIZER COMPLEXITY COMPARISON (NUMBER OF REAL MULTIPLICATIONS)

	$N=2$	$N=4$	$N=8$	$N=16$	$N=32$
LMS	32	128	512	2048	8192
FLMS	72	184	448	1056	2432

Table III shows that the FDE is much better than the TDE in terms of computational complexity when $N \geq 8$. These statistics confirm the FPGA implementation results that we got for FDE and TDE. As stated in the previous section, a 4-tap TDE consumes almost 80% of the FPGA resources. The 4-tap FDE needs over 200% of the FPGA resources, which confirms the theoretical results confirming that for $N < 8$, the LMS algorithm offers significant savings over the FLMS algorithm. Simulation results according to Simulink or SysGen/Simulink confirm that the FDE is more efficient than the TDE because it converges faster and second it reaches a lower error rate.

The design and implementation of domain equalizers has gained the attention of researchers recently. Dinis *et al.* [9] designed a FDE for a receiver system that is optimized for Offset Quaternary Phase Shift Keying. The authors of [9] have adopted the bit error rate (BER) as reference for performance evaluation. Mori *et al.* [10] presented a method to estimate the average block error rate performance of star 32/64-QAM schemes employing a FDE that is designed for orthogonal frequency division multiple access systems. The authors of [11] presented an ASIC hardware implementation of a FDE and measured power consumption and BER for their design using simulation. Mehana and Nosratinia [12], provided an analysis of a single-carrier FDE for cyclic delay diversity and Alamouti signaling schemes in order to obtain a threshold rate for the full spatial-temporal diversity. Finally, Li *et al.* [13] presented a sliding window FDE for multimode systems which operates on the time-domain received signal. The studies discussed above were concerned with performance parameters of the FDE such as SNR and BER. These parameters were estimated using Matlab. However, in all aforementioned methods, the error resulting from data conversion between different number domains is never measured or considered in the analysis. This error is due to handling the design and implementation of the equalizer at different levels of abstraction. Regardless of how small the error is, it can be amplified when introduced into an algorithm with repetitive and accumulative nature, such as FDE. The work we presented in this paper considers the TDE and FDE from this perspective and conducts performance analysis for the error resulted from data conversions for equalizer models at different levels of abstraction.

VI. CONCLUSIONS

We proposed a comprehensive method for the performance analysis of an equalizer implementations in time and frequency domains. The equalizer can be implemented using the LMS algorithm in time domain, or the FLMS algorithm in the frequency domain. In either implementation, a number of mathematical operations are performed on data in three different number domains: floating-point, fixed-point and real. Previous work in the signal processing field have shown that the FDE generally is more efficient than the TDE. However, the performance comparison has not been conducted in terms of error analysis at several levels of abstraction including different number domains. In addition, the efficiency of the different methods has not been considered in terms of hardware implementations in FPGA. The proposed methodology spreads over various design flow levels considering different number domains in addition to FPGA implementations of the algorithms. The conducted performance analysis compares between the two designs using two different metrics: error estimation and utilization of FPGA component in synthesize. We found out that the FDE design based on FLMS algorithm outperform the TDE. As a future work, we to formally model the time domain design, and then integrate it with the formal model of the frequency domain equalizer. This will enable conducting formal performance error analysis between the time and frequency domain implementations of the equalizer.

REFERENCES

[1] M. O. Fril, "Frequency Domain Adaptive Filtering," M.S. thesis, National University of Ireland, UK, 2005.

[2] A. Souari, A. Gawanmeh, S. Tahar, and M. Lassaad Ammari, "Design and Verification of a Frequency Domain Equalizer," *Microelectronics Journal*, vol. 45, no. 2, pp. 167–178, February 2014.

[3] Mathworks, "Matlab and Simulink. http://www.mathworks.com," 2014.

[4] Xilinx, "System Generator for DSP Reference Guide," 2008.

[5] John G Proakis, *Digital Communications*, McGraw-Hill, 1995.

[6] Lajos Hanzo, Soon Xin Ng, WT Webb, and T Keller, *Quadrature Amplitude Modulation: From Basics to Adaptive Trellis-coded, Turbo-equalised and Space-time Coded OFDM, CDMA and MC-CDMA Systems*, Wiley-IEEE Press, 2004.

[7] A. Souari, A. Gawanmeh, S. Tahar, and M. Lassaad, "Performance Evaluation of an a Equalizer in Frequency and Time Domains," Tech. Rep., Electrical and Computer Engineering Department, Concordia University, Montreal, Quebec, Canada, 2014. http://hvg.ece.concordia.ca/Publications/TECH_REP/DVFDE_TR13/.

[8] Xilinx, "http://www.xilinx.com," 2014.

[9] R. Dinis, M. Luzio, and P.Montezuma, "On the Design of Frequency-Domain Equalizers for OQPSK Modulations," in *IEEE Sarnoff Symposium*, 2010, pp. 1–5.

[10] C. Mori, T. Kawamura, N. Miki, and M. Sawahashi, "Link-level Performance of Star 32/64 QAM Schemes using Frequency Domain Equalizer for DFT-Precoded OFDMA," in *International Symposium on Wireless Personal Multimedia Communications*, 2012, pp. 266–270.

[11] K. Komatsu, S. Kameda, M. Iwata, S. Tanifuji, N. Suematsu, Tadashi Takagi, and K. Tsubouchi, "ASIC Implementation of Frequency Domain Equalizer for Single Carrier Transmission," in *General Assembly and Scientific Symposium*, 2011, pp. 1–4.

[12] A.H. Mehana and A. Nosratinia, "Single-Carrier Frequency-Domain Equalizer with Multi-Antenna Transmit Diversity," *IEEE Transactions on Wireless Communications*, vol. 12, no. 1, pp. 388–397, 2013.

[13] J. Li, E. Bala, and R. Yang, "Sliding Window-Frequency Domain Equalization for Multi-mode Communication Systems," in *IEEE Long Island Systems, Applications and Technology*, 2013, pp. 1–6.