Hardware Implementation of Narrowband PLC Channel Emulator Based on Multipath Channel Model Using the Frequency Domain Approach

Ann E. Dulay, Roderick Yap, Nicholas Thomas Ray P. del Castillo, Kayle C. San Juan, Maxilyn C. Tan
Electronics and Communications Engineering Department, De La Salle University, Manila, Philippines
ann.dulay@dlsu.edu.ph

***Abstract*—Narrowband PLC channel emulators are devices used to test PLC transceivers without the need for the latter to be connected directly to the hazardous and erratic power lines. It is also more cost effective and faster than simulators. The state-of-art NBPLC channel emulators have been implemented on the hardware using impulse response. While this is much simpler than the frequency domain approach, however, it is slower especially when it involves a large set of coefficients. For most designs, higher transform points would require a large size of coefficients, where normally, an external device like a PC or DSP is needed to generate the coefficients. In this paper, a standalone FPGA is used to implement the channel emulator by operating the emulator in the frequency domain.**

***Index Terms*—PLC; Emulator; Frequency Domain; Narrowband.**

I. INTRODUCTION

Power line communications (PLC) is the process of sending data over the power cables [3], [4]. PLC has been here since the 70's for the voice communications and teleprotection in remote power plants, and recently for smart grid applications [5]. The communications through the power lines between devices are made possible by PLC transceivers. Development of the PLC transceivers takes into account the channel characteristics of the power line network. Due to the stochastic nature of the PLC channel, testing of the PLC transceivers on actual and live power network is not only hazardous, it also does not guarantee reproducible results, hence, it becomes difficult to pinpoint the cause of the problem for cases where the transceiver does not successfully transmit or receive signals. PLC channel emulators can replace the real channel to provide reproducible and less hazardous testing process for PLC devices. The challenge, however, is to mimic as close as possible the real PLC channel. Several models have been developed to provide the closest response of the PLC channel, and thus far, the multipath channel model presented by Zimmermann and Dostert [6] has gained a lot of attention. Gotz [7] had used this model in their channel emulator but they used the impulse response for the channel transfer function whereby an external PC is needed to generate the filter coefficients. In this paper, the emulator is operating in the frequency domain. This approach eliminates the need for an external PC or device making the FPGA a standalone device.

II. PRIOR STUDIES

PLC channel emulators are very handy pieces of hardware that mimic the characteristics of the PLC channel so as to provide a convenient and fast way of testing PLC devices, either as part of the production process or to test new designs or even testing of an existing or new PLC standard. PLC channel emulators are classified according to the following: (1) frequency band - Narrowband or Broadband, (2) stochastic or deterministic model - multipath or transmission line models, (3) time domain or frequency domain processing, and (4) FPGA or PC-based [8], [9].

Most of the PLC channel emulators developed are for broadband PLC and uses time domain processing. This includes Gotz [10], and Sebeck [11]. Two papers that discuss PLC channel emulator for narrowband is given in [2] and [1]. Both studies show an emulator that implements impulse response for the channel transfer function. The implementation of the impulse response in hardware is implemented using a FIR filter. However, the difficulty lies in the generation of the filter coefficients. Due to this, a PC or a DSP is interfaced to the FPGA to generate the coefficients. In this paper, the transfer function is implemented in the frequency domain, thus, frequency domain convolution is carried out easily by simple multiplication of the signal and the transfer function thereby eliminating the additional hardware external to the FPGA.

III. DESIGN CONSIDERATIONS

Figure 1 shows the block diagram of the Narrowband PLC channel emulator. The ADC converts the analog signal to discrete values. The ADC used is LTC2389-18. It is an 18-bit high-speed SAR ADC that operates from a single 5 V power supply. It was configured as a pseudo-differential ADC converting voltage from 0 to 4.096 V. The Linear Regression module converts the ADC output to a correct numerical value that reflects the value of the analog input. The output of the linear regression is then passed on to the FFT module that transforms the discretized time domain data into the frequency domain. To emulate the typical power line, the channel transfer function module output is convolved with the FFT output. Convolution in the frequency domain is carried out by multiplying the two signals. The result is then added to a noise signal produced by the AWGN module. Although PLC noise is typically not Gaussian, AWGN is used to demonstrate the addition of noise in the frequency domain. The generated sum of the multiplier block and the AWGN block is fed to the inverse FFT module to bring back the signal to the time domain. The magnitude converter generates the magnitude part of the complex signal, then an

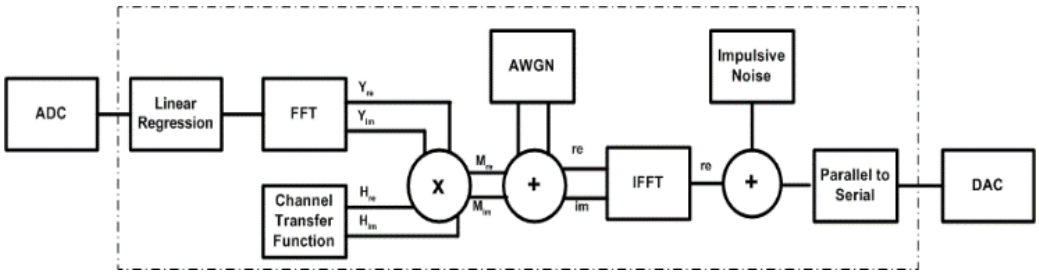


Figure 1: Block Diagram of the Narrowband PLC Channel Emulator

impulsive noise is added before finally converting it to serial data that is subsequently fed to the DAC module. The DAC module uses LTC2752 dual 16-bit serial output digital-to-analog converter. The total number of clock cycles required to complete the 16-bit conversion is 24 CP. The following subsections discuss the emulator blocks implemented in the FPGA.

A. Linear Regression

ADCs basically convert signals by generating a unique code for each step size that it resolves notwithstanding the actual value of the signal. The linear regression (LR) module corrects this by converting the ADC binary output to a number that represents the actual value using a 4,14 fixed point format. The first number represents 1 bit for the sign and 3 bits for the integer. The 3 bits is enough to accommodate the highest signal amplitude of 4.096 V. The 14 bits is for the fraction part. Table 1 below shows a portion of the values that are to be translated by the linear regression module. After tabulating all the digital equivalence of the analog input, a simple mathematical relation is obtained and is given in Equation (1).

Table 1
The First 10 of the 262,144 ADC Output and Conversion by LR

Analog Signal	ADC Output, x	Linear Reg out, y
1.91e-05	00000000000000001	00000000000000000
3.81e-05	00000000000000010	00000000000000000
5.72e-05	00000000000000011	00000000000000000
7.63e-05	00000000000000100	00000000000000001
9.54e-05	00000000000000101	00000000000000001
0.00011444	00000000000000110	00000000000000001
0.00013351	00000000000000111	00000000000000010
0.00015259	00000000000001000	00000000000000010
0.00017166	00000000000001001	00000000000000010
0.00019074	00000000000001010	00000000000000011

$$y = 0.256x - 0.752 \tag{1}$$

B. FFT Module

The signal from the Linear regression module is fed to the FFT block to convert the time domain signal to a frequency domain signal. The FFT module uses the Cooley-Tukey algorithm with the following mathematical function:

$$Y(k) = \sum_{n=0}^{N} y[n] e^{-j\frac{2\pi}{N}nk} \tag{2}$$

The FFT block is implemented in the XILINX core generator using Pipeline streaming I/O architecture which technically is a series of radix-2 butterflies. This architecture processes data continuously. The 18-bit real input to the FFT generates a complex output comprising of 28-bit real component and 28-bit imaginary component. The sampling Frequency is chosen to be 1MHz, this is, in fact, the smallest clock rate in the core. This is sufficient to sample the CENELEC narrowband PLC frequency range which is from 40 kHz to 90 kHz. The FFT transform points are chosen to be 512.

C. Channel Transfer Function

The channel transfer function is based on the multipath channel model developed by Zimmermann and Dostert [6]. Equation (3) shows the mathematical model of the transfer function.

$$H(f) = \sum_{i=0}^{N} g_i e^{-(a_0 + a_1 f^k) d_i} . e^{-j2\pi f \frac{d_i}{v_p}} \tag{3}$$

where the weighting factor g_i is a product of transmission and reflection factors (note that $g_i \leq 1$), is the number of paths, a0, a1 are the attenuation parameters, k is the exponent of the attenuation factor, d_i is the path length, and vp is the propagation velocity.

FPGA does not process imaginary numbers; hence, the above equation is converted to its rectangular form using Euler's theorem as follows:

$$H(re) = \sum_{i=0}^{N} g_i e^{-(a_0 + a_1 f^k) d_i} . \cos(2\pi f \frac{d_i}{v_p}) \tag{4}$$

$$H(im) = -\sum_{i=0}^{N} g_i e^{-(a_0 + a_1 f^k) d_i} . \cos(2\pi f \frac{d_i}{v_p}) \tag{5}$$

Take note that translation of these values in FPGA does not contain the imaginary number, j. However, the imaginary number j is taken into account during multiplication with other complex numbers which is the case when multiplying the transfer function and the FFT of the input signal.

Hardware modeling adopted for this transfer function was done using a look-up table. In [6], a constant value for the range of i has been provided for gi, a0, a1, di, k and vp . This leaves the transfer function to be evaluated for various frequency, f. A memory unit inside the FPGA has been created to hold the calculated values. The look up table includes both the real and imaginary components. Given the FFT points, the values in the look-up table must match the number of FFT points. Hence, the transfer function values stored in the LUT are based on the same set of frequencies calculated in the FFT operation and are generated from Matlab in order to simplify the process.

D. Multiplier

The multiplier module emulates the convolving of the transfer function and the input signal in the frequency domain. As has already been presented, both the transfer function and the FFT output generate complex values. The multiplier module presented here multiplies directly the numbers in their rectangular forms. The mathematical function of the operation is given as follows:

$$M(f) = (H_{re} + jH_{im})x(Y_{re} + jY_{im}) \quad (6)$$

where H_{re}, H_{im} are the components of the transfer function and Y_{re}, Y_{im} is the components of the FFT output, and M(f) = multiplier's output.

The above equation is written in the general format. However, it is possible for any of the components above to take negative values. Hence, this must be taken into account. A multiplier model that will give the correct result is developed based on the following mathematical operation:

Let $Hre + j\,Him = a + jb$ *and* $Yre + j\,Yim = c + jd$,

rewriting Equation (6) will give

$$M = (a + jb)\,x\,(c + jd) \quad (7)$$

This will generate the expression:

$$M = (ac + j2bd) + j(bc + ad) \quad (8)$$

Each of the product term **ac**, **bd**, **bc**, and **ad** are implemented on a multiplier that follows the same algorithm as shown in Figure 2. Let us take for instance the product term **ac**, where **a** is the multiplicand and **c** is the multiplier. In order to simplify the multiplication process, all the numbers must be converted to a positive value. Hence, each of the number goes through a sign converter. The sign converter generates the 2's complement of the number when the sign bit of the said number is 1, otherwise, the sign converter output merely copies the input. The outputs from the sign converter are then multiplied and the generated product is fed to the sign corrector module. The sign corrector module contains a comparator that compares the sign bit of the two numbers being multiplied such that when the numbers do not have the same sign bit, the sign corrector converts the output of the multiplier to its 2's complement format. However, a different sign corrector module is implemented for the bd product term in Equation (8) in order to take into account the effect of j^2 which is equal to -1. For this product term, the sign corrector module outputs the 2's complement when the sign bit of both numbers are the same.

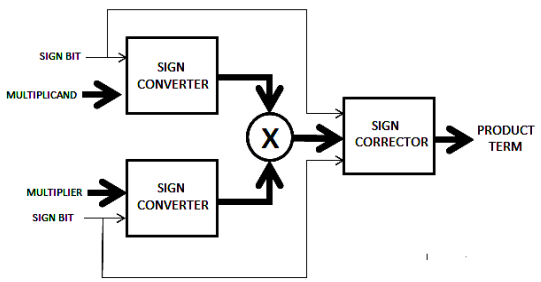


Figure 2: Multiplier Block

Referring to Equation (8), all real numbers and imaginary numbers are to be added separately to produce the final product term. Two adder modules are provided for this purpose.

E. Noise Generator

There are two noise generator modules included in the emulator, the AWGN module and the impulsive noise generator. The coefficients of the Gaussian noise are generated in Matlab and stored in an LUT. The impulsive noise generator is based on [10] and is also generated using Matlab. Both signals are added to the convolved signal, the AWGN in the frequency domain, and the impulsive noise in the time domain.

F. IFFT Module

The IFFT implemented in the FPGA is a Xilinx LogiCORE IP FFT that implements the Cooley-Tukey algorithm. It is basically the same FFT process except that the inverse FFT (IFFT) is computed by conjugating the phase factors of the corresponding forward FFT. The IFFT function is given by:

$$y[n] = \frac{1}{N}\sum_{k=0}^{N} Y[k]e^{j\frac{2\pi}{N}nk} \quad (9)$$

The input to the IFFT is the 28 bits generated by the multiplier and resulting output is truncated to 28 bits. The Xilinx LogicCore uses the same GUI for the IFFT and FFT functions. The function is differentiated only by changing the value of fwd_inv we option in the GUI, whereby it is set to 1 for FFT function and 0 for IFFT. The 1/N factor is a scaling factor to be able to reconstruct the signal correctly.

IV. DATA AND RESULTS

The Verilog code of every design module was synthesized and simulated using XILINX ISIM. The result of ISIM simulation is then compared with MATLAB simulation to verify accuracy. Both simulations revealed similar results for all generated complex number values. The complete Verilog code representing the power line channel emulator is then implemented on a VIRTEX 5 FPGA. To test the performance of the FPGA, a sine wave input is fed to it using an Analog to Digital Converter. The generated output from the FPGA is converted back to analog using a Digital to Analog Converter (DAC). The FPGA was tested for various scenarios. The following subsections discuss each of these test scenarios and the corresponding outputs. For all the scenarios, the ADC and DAC clocks are 24:1 ratio owing to the serial processing of the DAC. The clock rates used for the following test set-ups are 137.5 kHz for ADC and 3.3 MHz for DAC.

A. Flat Power Spectrum of FPGA

In order to determine if the FPGA is not contributing to the overall errors that the external signal is subjected to, a flat power spectrum test is done. Figure 3 shows that the power spectral density is small compared to the input signal. Hence, the noise contributed by the FPGA can be considered insignificant.

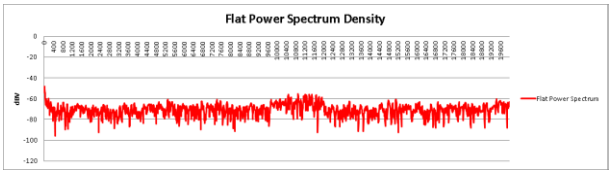


Figure 3: Flat Power Spectrum of the FPGA Channel Emulator

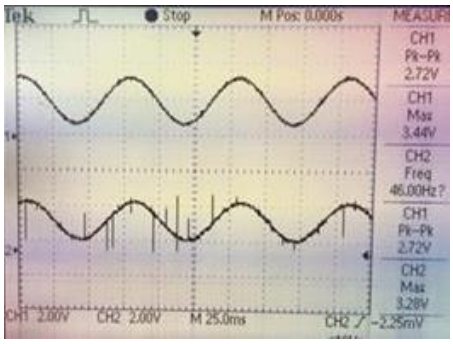


Figure 4: Input-Output Pair for 46 Hz Input sine wave at TF=1

B. Frequency Domain Test

The purpose of this test is to determine if the FPGA is able to process correctly the multiplication process of the FFT signal and the transfer function. The noise LUTs are disabled in this test. Figure 4, 5 and 6 show the input-output pairs generated for various input frequencies at a transfer function set to 1. This is done to easily view the waveform since the output simply follows the input. At 1 kHz, the output signal is already heavily distorted.

In another test, the transfer function is set to 0.8 with the noise generators still disabled. Figure 7 shows the corresponding input-output waveforms for (a) 11 Hz and (b) 1 kHz respectively. As expected, the output is 0.8 times the input but with the 1 kHz signal having noticeable distortion.

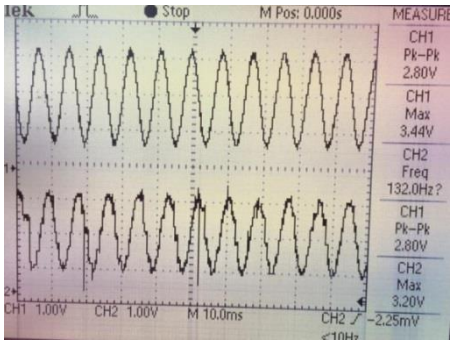


Figure 5: Input-Output Waveforms for 132 Hz Sine Input at TF=1

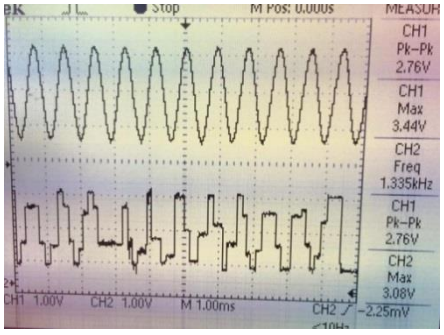
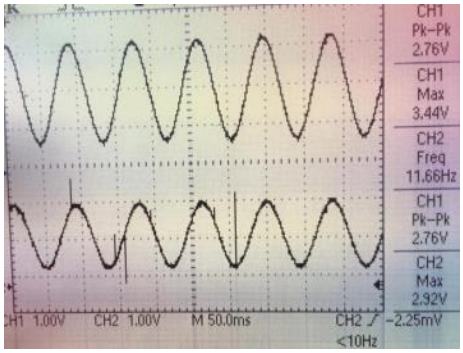
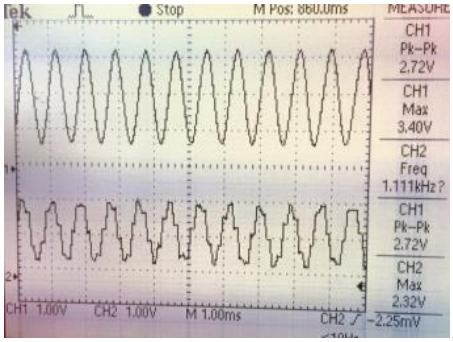


Figure 6: Input-Output Waveforms for 1.33~kHz Sine Input at TF=1



(a)

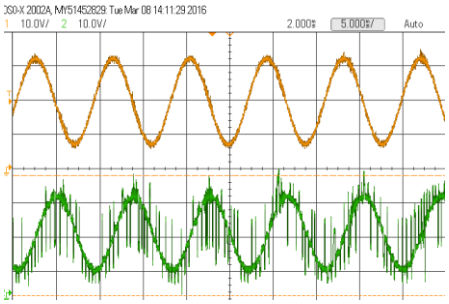


(b)

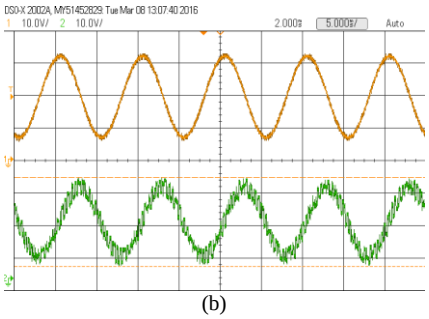
Figure 7: Input vs Output at a TF=0.8 for (a) 11 Hz input (b) 1kHz input

C. Noise Test

In the following tests, noise is added to the convolved signal. The first noise added is an AWGN noise. Figure 8 shows the output waveforms diluted with 100µV (Figure 8(a)) and 100mV (Figure 8(b)) white noise. The transfer function for all these tests is set to 1 to easily view the result.



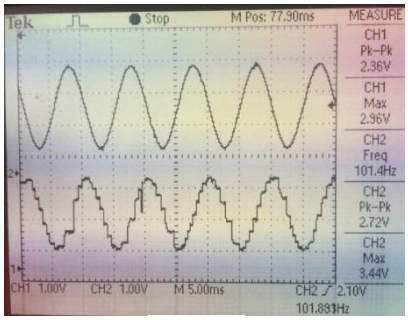
(a)



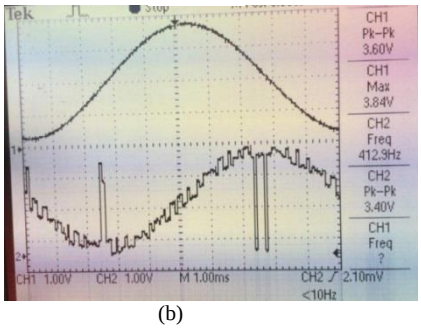
(b)

Figure 8: Input vs Output Waveforms with (a) 100µV AWGN (b) 100mV AWGN Added

Figure 9(a) shows the output which results when an impulse is added to the signal. The last test involves the addition of both the impulse and AWGN to the signal. Figure 9b shows the generated output by the DAC when all the signals are added.



(a)



(b)

Figure 9: Input vs Output Waveforms with (a) Only Impulse Added (b) Both Noises Added

V. CONCLUSION

The narrowband PLC channel emulator is successfully implemented on Virtex-5 FPGA using the frequency domain approach. Convolving of the transfer function response and the transformed signal is done using simple multiplication. Since the Zimmermann multipath contains phase information thereby producing complex numbers, the Euler's theorem is used to convert the polar form to rectangular form allowing the separate processing of real numbers from imaginary numbers. The LTC2389-18 ADC is configured to resolve the signal by 18-bits on a pseudo-differential mode. The LTC2752 16-bit DAC converted back the signal to analog at a clock rate which is 24 times faster than the ADC due to the serial processing of the data from the FPGA. The FFT operation is working at a target clock rate of 1MHz at 512 FFT points in order to process narrowband PLC signals up to 90 kHz. When implemented on the Virtex-5 FPGA, the entire design used up around 5% of the total Block RAM and around 75% of the DSP logic elements.

ACKNOWLEDGEMENT

The authors would like to thank the University Research Coordination Office (URCO) and Science Foundation of De La Salle University for funding this research.

REFERENCES

[1] M. Bauer, W. Liu and K. Dostert, "Channel emulation of low-speed plc transmission channels," in *IEEE International Symposium on Power Line Communications and Its Applications, 2009. ISPLC 2009* , March 2009.

[2] W. Liu, *Emulation of narrowband powerline data transmission channels and evaluation of PLC systems,, Ph.D. dissertation*, Karlsruher Instituts fur Technologie, 2013.

[3] K. Dostert, Power Line Communications, New Jersey: Prentice Hall, 2001.

[4] Cypress, "What is power line communication?," EE Times, August 2011. [Online]. Available: http://www.eetimes.com/document.asp?doc id=1279014.

[5] S. Galli, A. Scaglione and Z. Wang, "For the grid and through the grid: The role of power line communications in the smart grid," in *Proceedings of the IEEE*, June 2011.

[6] M. Zimmermann and K. Dostert, "A multipath model for the powerline channel," *Communications, IEEE Transactions on*, vol. 50, no. 4, p. 553–559, Apr 2002.

[7] M. Gotz and K. Dostert, "A universal high speed powerline channel emulation system," *Broadband Communications, 2002. Access, Transmission, Networking. 2002 International Zurich Seminar on*, p. 24–1–24–6, 2002.

[8] W. Zhu, X. Zhu, E. Lim and Y. Huang, "State-of-art power line communications channel modeling echnology and Quantitative Management,," in *Procedia Computer Science, First International Conference on Information*, 2013.

[9] V. Oksman and J. Zhang, "G.HNEM: the new ITU-T standard on narrowband PLC technology," *Communications Magazine*, vol. 49, p. 36–44, December 2011.

[10] F. Canete, L. Diez, J. Cortes, J. Sanchez-Martinez and L. Torres, "Time-varying channel emulator for indoor power line communications," in *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*, Nov 2008.

[11] M. Sebeck and G. Bumiller, "Power-line analysing tool for channel estimation, channel emulation and noise characterisation," in *Proceedings of the 5th International Symposium on Power Line Communications and Its Applications*, 2001.

[12] "Plc channel model," Working Group on Power Line Communications, University of Malaga, [Online]. Available: http://www.plc.uma.es/channels.htm. [Accessed July 2016].