FPGA-based Optical Kerr Effect Emulator

Downloaded from: https://research.chalmers.se, 2022-08-11 03:21 UTC

Citation for the original published paper (version of record):

N.B. When citing this work, cite the original published paper.
FPGA-based Optical Kerr Effect Emulator

Keren Liu\textsuperscript{1}, Erik Börjeson\textsuperscript{1}, Christian Häger\textsuperscript{2}, and Per Larsson-Edefors\textsuperscript{1}

\textsuperscript{1}Dept. of Computer Science and Engineering, Chalmers University of Technology, Gothenburg, Sweden
\textsuperscript{2}Dept. of Electrical Engineering, Chalmers University of Technology, Gothenburg, Sweden
perla@chalmers.se

Abstract: We propose a digital emulator of the optical Kerr effect, suitable for FPGA implementation. In addition, we study a combined PMD and Kerr emulator implementation with respect to DSP hardware aspects such as fixed-point performance. © 2022 The Author(s)

1. Introduction

In fiber-optic communication, there are several impairments that affect the BER performance at the receiver. The linear polarization-mode dispersion (PMD) and the nonlinear optical Kerr effect have a combined detrimental effect on the transmitted signal \cite{1}, but this effect can be complex to study as PMD varies over time. To thoroughly assess the impact of the combined PMD-Kerr effect on a receiver DSP implementation, one option is to conduct optical transmission experiments. However, besides requiring expensive equipment, experimental setups can be challenging to integrate with real-time DSP and often lack precise control of the underlying channel parameters.

The Fiber-on-Chip approach \cite{2} offers an alternate way to perform real-time analysis of a fiber-optic communication system. Here, a digital system model runs in real-time on an FPGA, making real-time control and analysis straightforward. This paper extends our previous work in \cite{2,3} and proposes a digital emulator of the optical Kerr effect that, in combination with the PMD emulator in \cite{3}, forms a real-time PMD-Kerr emulator that is suitable for implementation on an FPGA.

2. Digital Emulation of Optical Kerr Effect

Our digital Kerr effect emulator is based on a numerical solution to the Manakov-PMD equation \cite{4}, in which a fiber is divided into multiple PMD sections. Each section’s Kerr effect is modeled as \( \hat{u} = u \exp(i\gamma\|u\|^2) \), where \( u = [u_x(t,z), u_y(t,z)] \) is the Jones vector of the complex baseband signals in the \( x \) and \( y \) polarizations, \( t \) and \( z \) are the propagation time and distance, respectively, and \( \gamma = \frac{8}{9} \gamma \) represents the Kerr parameter \( \gamma \) multiplied by the section length \( L \) and the averaging coefficient \( \frac{8}{9} \). This model can be further written as

\[
\begin{align*}
\hat{u}_{xi} &= u_{xi} \cos(\phi) - u_{xq} \sin(\phi) \\
\hat{u}_{xq} &= u_{xi} \sin(\phi) + u_{xq} \cos(\phi)
\end{align*}
\]

where \( u_x = u_{xi} + j u_{xq} \) and \( u_y = u_{yi} + j u_{yq} \) are the inputs and \( \Box \) denotes the outputs.

Fig. 1 shows a block diagram of the digital Kerr effect emulator, which is pipelined to balance the timing paths and increase the clock speed. A look-up table (LUT), which comprises all sine values, is indexed by the input angle \( \phi \) with a range of \([0, \pi/2]\). The Limit block converts the angle to the range \([0, \pi/2]\).

3. Emulator Structure

The combined PMD-Kerr emulator developed in this paper uses a transmitter from the CHOICE environment \cite{5,6} and PMD emulator components developed in \cite{3}. Fig. 2 illustrates the system structure, inside which emulation of the combined impacts of PMD and the Kerr effect is realized. The pseudo-random data sequence used for transmission over two polarizations is modulated to QPSK before being upsampled to two samples per symbol and convolved with a 51-tap root-raised-cosine (RRC) filter with a roll-off factor of 0.1.

\[
\text{Fig. 1: Block diagram of the Kerr effect emulator. Registers are represented by } z^{-1}. \text{ The boxed rotation block is shown only for the } x \text{ polarization, and the rotation block for the } y \text{ polarization is a duplication of the shown one. The inputs to the rotation block are also delayed by three } z^{-1} \text{ which are not shown.}
\]
The digital PMD model consists of multiple concatenated sections [3] and the Kerr emulator is inserted in each section after the PMD emulator. Each PMD section contains one Rotation and one Delay component. The former rotates the x and y polarization with an angle of $\theta_k$ in section $k$, while the latter utilizes an $n$-tap Lagrange fractional-delay filter (similar to [7]) to apply a differential group delay (DGD) of $\tau$ between the two polarizations. Here, $\tau$ is a fraction of the symbol period ($T$). The rotation angle $\theta_k$ can be time varying and different for each section, while the DGD $\tau$ and the Kerr parameter $\gamma$ are kept the same in all sections for simplicity.

4. Analysis of Performance and Resource Utilization

We first perform floating-point MATLAB simulations to compare the time-domain (TD) PMD-Kerr model, which uses an $n$-tap Lagrange filter to realize DGD, with a reference frequency-domain (FD) PMD-Kerr model, which uses a DGD based on FFT/IFFT. The system assumptions are $\overline{\gamma} = 1.4$ rad/W (assuming $\gamma = 1.4$ rad/W/km and $L = 100$ km) and an overall transmitted power of 0 dBm. The $\theta_k$, $k = 1, 2, ..., K + 1$ is randomly generated with a uniform distribution on $[-\pi, \pi]$ and is kept unchanged during the simulations. Fig. 3 shows the results for a 16,384-sample simulation, where $n$ is varied from 5 to 15, while $\tau \in \{0.06T, 0.24T\}$ and $K \in \{1, 5, 10\}$. The error is calculated as $\overline{\gamma} = \frac{1}{n} \sum_{\tau} \log_{10}(\text{error})$.

To further evaluate our proposed emulator, we use logic simulations to compare the fixed-point implementation against a TD MATLAB implementation, with $\overline{\gamma} = \frac{1}{n} \sum_{\tau} \log_{10}(\text{error})$ for $K = 10$ and a 5-tap Lagrange filter. The wordlengths $a, b, c, d$ and $e$ are individually varied from 8 to 16: $a$ corresponds to the data sample $x_0, x_1, x_2, y_1$, and $y_2$, $b$ to the Kerr parameter $\gamma$, $c$ to the rotation angle $\theta_k$ and the Kerr angle $\phi$, $d$ to the FIR taps of the Lagrange filter, and $e$ to the RRC taps. All parameters are represented in signed fixed-point format. To average the rounding error caused by the different values of $\theta_k$, for each symbol in the simulation, i.e., two samples, a new set of $\theta_k$, $k = 1, 2, ..., K + 1$ is randomly generated with a uniform distribution on $[-\pi, \pi]$. Fig. 4 shows the results for a simulation with 16,384 samples.

To analyze the resource utilization, we synthesized the system in Fig. 2 to a Xilinx VC709 development board with a 100 MHz clock. We choose 12-bit wordlengths for all signals and use section counts $K$ between 1 and 10. As can be expected, the usage of LUT and DSP resources grows linearly with an increasing $K$. As Table 1 shows for $K = 10$, the DSP slices are the bottleneck and this 10-section system utilizes 26% of the available DSPs, which leaves enough room for implementing an equalizer on the same FPGA.

**Table 1: Resource utilization on Xilinx VC709.** The $K$ entry includes all Kerr modules in the system. The values are given as number of used blocks (ratio of used blocks to available blocks).

**References**


