FPGA Implementation of Signal Processing Algorithms in Coherent Optical Systems

N. M. Pinto, L. M. Pessoa, J. C. Ferreira and H. M. Salgado

***Abstract*—This paper describes the FPGA hardware implementation of well known algorithms for signal estimation in coherent optical systems. The results address both serial and parallel implementations, in terms of complexity and performance. Proof of principle results were obtained, which could be easily scaled to currently used optical systems data rates.**

***Index Terms*—Coherent Systems, Chromatic Dispersion Compensation, FPGA.**

I. Introduction

COHERENT optical communications carry several advantages over intensity modulated direct detection systems, namely the ability to use phase modulated (M-QPSK) and multi-level constellations (M-QAM), due to the preservation of the electric field from the optical domain to the electrical domain, provided the sampling is at least at Nyquist rate. Additionally, it enables quasi-exact compensation of linear transmission impairments, such as chromatic dispersion (CD) and polarization mode dispersion (PMD) by a linear filter [1], which can operate adaptively to overcome time-varying impairments. These systems have gained renewed interest due to the availability of high speed digital signal processing, which allows for complex operations to be carried out in the digital domain, enabling high potential for a reconfigurable software defined optical receiver.

AD converters will be able to satisfy the required high sampling rates in the near future for optical long-haul high speed transmission systems. Moreover, as soon as sufficiently high speed data converters are available, field programmable gate arrays (FPGA) are a very flexible implementation platform. Considering that the Local Oscillator (LO) phase needs to be locked to the signal phase, to avoid the difficulties associated with the Optical Phase Locked Loop (OPLL), the synchronization can be done in the DSP, by digital phase estimation techniques, allowing for a free running LO. Algorithms suitable for phase estimation and dispersion compensation have been studied in [2]. The implementation of these algorithms using FPGAs, capable of parallel processing [3] are discussed in this paper. We implement the adaptive dispersion compensating algorithms using the *System Generator* tool from Xilinx. Section II details adaptive equalization structures, followed by a description of the FPGA implementation. Sections IV and V present the results and conclusions.

N. M. Pinto, L. M. Pessoa, J. C. Ferreira and H. M. Salgado are with INESC Porto, Faculdade de Engenharia da Universidade do Porto, Porto, Portugal (email: ee03171@fe.up.pt, luis.pessoa@ieee.org, jcf@fe.up.pt, h.salgado@ieee.org).

II. Adaptive Equalization

A. Constant Modulus Algorithm - CMA

The CMA is the most used algorithm for adaptive equalizers, essentially because of its robustness and ability to converge prior to phase recovery. The signal at the equalizer output is obtained through convolution of the equalizer coefficients with the digitized signal.

$$y(n) = \mathbf{w}^H(n) \cdot \mathbf{u}(n) \tag{1}$$

Then the error signal is calculated as follows:

$$e(n) = y(n) \cdot (R_2 - |y(n)|^2) \tag{2}$$

where R_2 is a constant dependent on the selected constellation. The coefficient update is then given by:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu \cdot \mathbf{u}(n)e^*(n) \tag{3}$$

where μ is the algorithm step size. It is important to consider the initialization of the coefficients for successful convergence. A center spike initialization is required, where the central coefficient is equal to the unity, while all other coefficients are set to zero. This algorithm provides good performance, but disregarde the signal phase, causing the constellation to twist. On the other hand, the next algorithm takes the phase into account.

B. Least Mean Squares - LMS

The LMS is very similar to the CMA, except for the signal error calculation, where an extra module is used to calculate symbol decisions at the equalizer output. The error can the be calculated as:

$$e(n) = d(n) - y(n) \tag{4}$$

where $d(n)$ is the decided symbol. The LMS initialization might be done in two ways. A training sequence is used for initial convergence, the LMS being switched to Decision Directed (DD) mode thereafter. However as training sequences are difficult to provide, a more elegant approach can be used, where the LMS coefficients are initialized to the coefficients obtained after CMA convergence. This is the approach taken in this work.

III. FPGA implementation

In order to implement the equalizers in FPGA, the *System Generator* tool was used, which is a *Matlab* based platform. *Xilinx* libraries are added to *Simulink*, which contain several

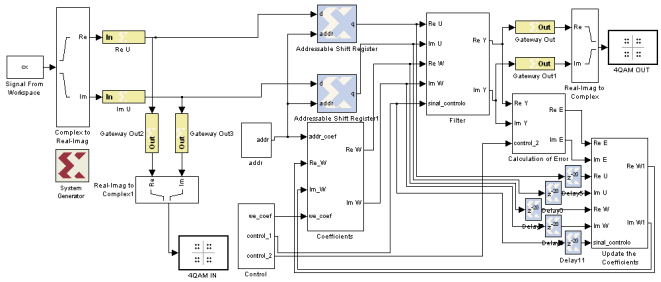


Fig. 1. CMA equalizer implemented in System Generator. The FPGA is delimited by blocks *In* - Input path - and *out* - Output path.

efficient blocks that can be used to accomplish the required processing.

Figure 1 shows a block diagram of the CMA algorithmimplemented in *System Generator*. The real and imaginary parts of the signal are treated separately. An important issue is the precision used in the signal digitalization. In this work 18 bits were used, split in the following way: 1 bit for the signal, 5 bits for the integer part and 12 bits for the fraction part. As illustrated, the algorithm stores both coefficients and data. The number of coefficients used is 13 [4], which is also the number of consecutive stored results. The filter performing the operations of equation (1) follows. The calculation of the error given in equation (2) is then executed, after which the coefficient update given in equation (3) is accomplished. The LMS was also implemented, with the previously mentioned difference in the symbol decision. A sequential and a parallel version were implemented for each algorithm.

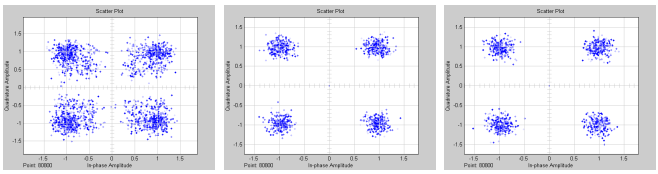
TABLE I
COMPARISON BETWEEN SEQUENTIAL AND PARALLEL MODEL

Components	**Serial**	**Parallel**
Slices	1070	6944
FFs	939	5285
LUTs	1758	9933
IOBs	108	108
Emb. Mults.	14	60
Characteristics		
Max. Frequency	57.917 MHz	97.144 MHz
Min. Period	17.266 ns	10.294 ns
Minimum timings		
Calculus duration	26 periods	10 periods
Symbol duration	448.916 ns	102.94 ns
Throughput	2.2 MSymbols/s	9.7 MSymbols/s

The table above shows the hardware resource allocation, timings and data rates, for the Virtex 4 (xc4vfx60-10ff672). The parallel version occupies 6x more resources than the serial version while allowing for approximately 5x faster operation. The greatest advantage of using *System Generator* is that the same implementation is still valid for a faster FPGA with more resources, requiring only the generation of a new Netlist. Another kind of parallelism can be used, by having several LMS or CMA units perform convolution at the same time, as long as each set of samples of the same symbol (2 samples in this work) is forwarded to the same unit.

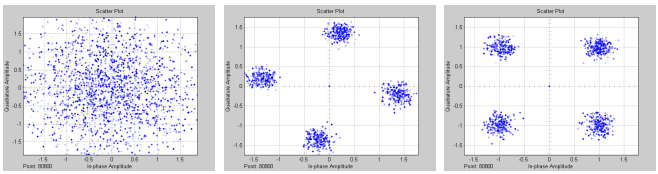
IV. RESULTS

In our numerical simulations, we have used a NRZ pulse shape, obtained by passing an ideal rectangular pulse train through a 5th-order low pass Bessel filter with a 3-dB bandwidth of 80% of the symbol-rate. For the anti-alias filter, a 3rd-order low pass Bessel filter was used. We present the results of a 4-QAM constellation, with and without CD for both algorithms (CMA and LMS).



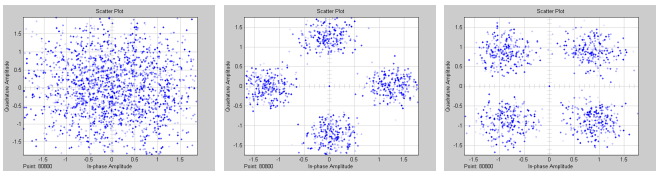
(a) Equalizer input (b) CMA (c) LMS

Fig. 2. 4QAM constellation without dispersion.



(a) Equalizer input (b) CMA (c) LMS

Fig. 3. 4QAM constellation with 200 km fiber.



(a) Equalizer input (b) CMA (c) LMS

Fig. 4. 4QAM constellation with 500 km.

Figure 2 plots the CMA and LMS performance without dispersion. Figure 3 shows that the presented algorithm successfully compensates CD of 200 km of standard single mode fiber (SSMF), since the 13 taps are capable of synthesizing the inverse transfer function of the fiber CD [4]. For 500 km (Figure 4) the performance is degraded. An additional equalization module with fixed coefficients obtained from the inverse *Fourier* transform of the fiber transfer function would be required for distances above 200 km.

V. CONCLUSIONS AND FUTURE WORK

The implementation of adaptive equalization algorithms was carried out in a FPGA platform. We conclude these algorithms might be implemented with good performance given that a sufficient number of bits is used. We have successfully initialized the LMS algorithm with the coefficients resulting from CMA convergence. Furthermore, for high fiber distances an equalization module with fixed coefficients would be required. The implementation of a phase estimation algorithm proposed in [2] is the next step for this work.

REFERENCES

[1] E. Ip, et. al, "Coherent Detection in Optical Fiber Systems", Optics Express, pp. 753-762, Jan. 2008.

[2] L. M. Pessoa, et. al, "Efficient Implementation of a Phase Estimation Algorithm in Coherent Optical Systems", IEEE LEOS STM 2008, Mexico, July 2008.

[3] A. Leven, et. al., "Real-time implementation of 4.4 Gbit/s QPSK intradyne receiver using field programmable gate array", Electronics Letters, vol. 42, no. 24, Nov. 2006.

[4] S. J. Savory, "Digital filters for coherent optical receivers", Optics Express. vol. 16, no.2, Jan. 2008