COMPARISON OF FPGA EQUALIZER IMPLEMENTATIONS FOR HIGH-SPEED DATA TELEMETRY

Daniel Schmalz, Joseph Lennon, Enkuang Wang, Timothy Brothers, Ph.D.
Georgia Tech Research Institute
Atlanta, GA, 30318
{Daniel.Schmalz, Joseph.Lennon, Enkuang.Wang, Timothy.Brothers}@gtri.gatech.edu

ABSTRACT

This paper examines the real-time implementation of equalization techniques. Telemetry RF channels are formidable due to the nature of desert test ranges – specifically due to multipath, changing path loss from environmental effects, and thermal distortions. This challenge is further complicated by the high velocity nature of test assets. Optimization of channel equalization in a real-time scenario is essential for high speed data telemetry over extended distances. This paper examines the mathematical background of equalization techniques and presents results based on FPGA implementations. The results were obtained from Vivado High Level Synthesis (HLS), which generates HDL from C/C++, as well as traditional VHDL coding. The contribution to the state of the art in this paper is the determination of the technological maturity of HLS versus traditional hand coding and the comparison of FPGA implementations of equalization algorithms against current platforms.

INTRODUCTION

Equalization is needed in communications systems affected by channel distortions including intersymbol interference (ISI). ISI is particularly common in single carrier (SC) communication systems since symbol durations must decrease to lengths potentially shorter than the channel coherence time when a high data rate is desired. Recently, interest has been revived in SC communication because it offers small peak-to-average power ratio (PAPR) and higher robustness to frequency offset and phase noise. Additionally, SC modulations can reach comparable capaicities as multicarrier by first applying a linear filter, then cancelling out remaining interference [1].

A specific environment that depends on equalization is that of aeronautical telemetry. The required data rates in combination with the multipath dominated environment results in non-negligible ISI. Furthermore, due to the small weight and power (SWAP) constraints, multicarrier communications are often not practical [2]. A waveform that has appeared in recent telemetry standards is a variant of shaped offset quadrature phase shift keying (SOQPSK), called SOQPSK-TG [3]. With a constant envelope and well-contained spectrum usage, SOQPSK-TG has demonstrated potential for such applications [4].

Table 1: Notation of Symbols

Symbol	Description
L_1, L_2	length of data before and after time 0, respectively
L	equals $L_1 + L_2 + 1$, the total number of data samples
N_1, N_2	length of channel impulse response before and after time 0, respectively
N	equals $N_1 + N_2 + 1$, the total number of samples in the channel impulse response support
$\boldsymbol{x}$	vector of transmitted data of size L
$\hat{\boldsymbol{x}}$	estimation of transmitted data of size L
$\boldsymbol{y}$	vector of received data of size L
$\boldsymbol{w}$	vector of additive noise of size L
$\boldsymbol{h}$	vector of channel impulse response of size N
$\boldsymbol{c}$	vector of filter taps of size N
$\boldsymbol{H}$	matrix for applying channel impulse response – can be for circular or linear convolution depending on the context
$\boldsymbol{C}$	matrix for applying filter taps – can be for circular or linear convolution depending on the context
$\boldsymbol{F}$	unitary DFT matrix
$\boldsymbol{\Lambda_c}$, $\boldsymbol{\Lambda_h}$	diagnoal matrices containing the eigenvalues of $\boldsymbol{C}$ and $\boldsymbol{H}$ respectively
λ_c^i, λ_h^i	ith eigenvalue of $\boldsymbol{C}$ and $\boldsymbol{H}$ respectively
$\boldsymbol{e_i}$	vector containing an element equal to 1 at index i and elements equal to zero at all other locations
$[\cdot]_i$	this operator extracts the ith element of a vector
$(\cdot)^H$	this operator denotes conjugate transpose
$(\cdot)^T$	this operator denotes non-conjugated transpose

This paper demonstrates zero-forcing, MMSE, and CMA channel estimation and equalization of SOQPSK-TG data. The novelty relative to previous equalization papers such as [5] is implementation with field programmable gate arrays (FPGAs) instead of GPUs. Specifically, FPGA implementations are created using a combination of high level synthesis tools (HLS) as well as hand coded VHDL. To measure performance in typical aeronautical telemetry distortions, the result is tested on data corrupted by channels measured at Edward's Air Force Base (EAFB) [6] and additive white gaussian noise (AWGN).

The notation used in the following sections is listed in Table 1.

MODEL OF CHANNEL

The effect of a channel can be modeled using linear convolution with an impulse response and additive noise. Below, denote $\boldsymbol{x} \in \mathbb{C}^L$ as the transmitted data with indexed elements written as $x[n]$, $\boldsymbol{y} \in \mathbb{C}^L$ and $y[n]$ as the received data, $\boldsymbol{h} \in \mathbb{C}^N$ and $h[k]$ as the channel impulse resonse, and $\boldsymbol{w} \in \mathbb{C}^L$ and $w[n]$ as a AWGN noise. Let $N = N_1 + N_2 + 1$ and $L = L_1 + L_2 + 1$ with $L \gg N$.

$$y[n] = \sum_{k=-N_1}^{N_2} h[k]x[n-k] + w[n] \qquad \forall n \in \{-L_1 \dots L_2\} \tag{1}$$

Note that equation (1) requires knowlege of $x[-L_1-N_1] \dots x[-L_1-1]$ and $x[L_2+1] \dots x[L_2+N_2]$. In linear convolution, these values are assumed to be zero. An alternative option is to assume periodic boundary condtions

$$x[n] = x[(n + L_1 \text{ modulo } L) - L_1] \tag{2}$$

With this, equation (1) can be written as a matrix multiplication of $\boldsymbol{x}$ with a circulant matrix $\boldsymbol{H} \in \mathbb{C}^{L \times L}$. Below, the rows of $\boldsymbol{H}$ are circular shifts of $\boldsymbol{h_L} = [h[N_2] \dots h[-N_1] \quad 0 \dots 0] \in \mathbb{C}^L$.

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{w} \tag{3}$$

With the structure of $\boldsymbol{H}$, its spectral decomposition contains the unitary DFT matrix $\boldsymbol{F}$. Let $\boldsymbol{\Lambda_h}$ be a diagonal matrix containing the eigen values of $\boldsymbol{H}$ denoted by λ_i^h for index i.

$$\boldsymbol{H} = \boldsymbol{F^H \Lambda_h F} \tag{4}$$

The matrix $\boldsymbol{\Lambda_h}$ can be computed by taking the FFT of $\boldsymbol{h_L}$.

FREQUENCY BASED ZERO-FORCING EQUALIZER

The zero forcing equalizer is a linear filter $\boldsymbol{c}$ that satisfies

$$\boldsymbol{h} * \boldsymbol{c} = \boldsymbol{e_{N_1+1}} \tag{5}$$

where $\boldsymbol{e_{N_1+1}}$ is the unit vector whose $N_1 + 1$ component (the index corresponding to time zero) equals one while all other components equal zero. In the FFT domain under periodic boundary condtions, this can be computed as in equation (6) [7].

$$[\boldsymbol{Fc}]_i = 1/[\boldsymbol{Fh}]_i \tag{6}$$

In practice, (6) cannot be used since the channel is not known. Instead, $\boldsymbol{c}$ must be computed from the data. To do so, first define matrix $\boldsymbol{C}$ for applying the circular convolution of a vector with $\boldsymbol{c}$ similar to how $\boldsymbol{H}$ is defined for equation (3). Equations (5) and (6) imply that in the absence of noise, the following equation holds when $\boldsymbol{H}$ is full rank.

$$\boldsymbol{Cy} = \boldsymbol{x} \tag{7}$$

Suppose a receiver has data $\boldsymbol{y}$ for a known $\boldsymbol{x}$. To satisfy equation (7) in least squares sense, the following optimization problem must be solved.

$$\min_{\boldsymbol{c}} ||\boldsymbol{C}\boldsymbol{y} - \boldsymbol{x}||_2^2 \tag{8}$$

Note that there is no error in assuming circular convolution if a periodic training sequence is transmitted. Using the spectral decomposition of $\boldsymbol{C}$ and properties of the Euclidean norm, the objective function can be written as:

$$||\boldsymbol{C}\boldsymbol{y} - \boldsymbol{x}||_2^2 = ||\boldsymbol{F}^{\boldsymbol{H}}\boldsymbol{\Lambda}_{\boldsymbol{c}}\boldsymbol{F}\boldsymbol{y} - \boldsymbol{x}||_2^2 \tag{9}$$

$$= ||\boldsymbol{\Lambda}_{\boldsymbol{c}}\boldsymbol{F}\boldsymbol{y} - \boldsymbol{F}\boldsymbol{x}||_2^2 \tag{10}$$

Let $\boldsymbol{y_p} \in \mathbb{C}^N$ be received data of a known periodic pilot sequence. Suppose $\mathcal{I} \subseteq \{0 \dots N\}$ is the set of indices where $\boldsymbol{F}\boldsymbol{y}$ is non-zero. The solution below yields estimated eigen values $\hat{\lambda}_i^c$.

$$\hat{\lambda}_i^c = \begin{cases} [\boldsymbol{F}\boldsymbol{x}]_i / [\boldsymbol{F}\boldsymbol{y}]_i & i \in \mathcal{I} \\ 0 & i \notin \mathcal{I} \end{cases} \tag{11}$$

The filter can be applied to new data through elementwise multiplication in the DFT domain or by taking the IFFT of the vector $\hat{\boldsymbol{\lambda}}^{\boldsymbol{c}}$ to create a linear filter in the time domain. For the DFT domain approach, let b be the block index.

$$\boldsymbol{y_b} = y[bN, \dots, (b+1)N] \tag{12}$$

$$\hat{\boldsymbol{x}}_{\boldsymbol{b}} = \boldsymbol{F}^{\boldsymbol{H}}(\hat{\boldsymbol{\Lambda}}_{\boldsymbol{c}}(\boldsymbol{F}\boldsymbol{y_b})) \tag{13}$$

In the case that λ_i^h are close to zero, their corresponding λ_i^c (which ideally equals $1/\lambda_i^h$) will become very large. As a consequence, noise that has correlation with corresponding eigenvectors will be amplified to undesirable levels. The degree to which this effect takes place can be quanitified by the condition number of $\boldsymbol{H}$, which measures how "invertible" it is. Let $\kappa(\cdot)$ denote the condition number of a matrix, $\sigma_i(\cdot)$ the *i*th largest singlar value, and r_H the rank of matrix $\boldsymbol{H}$ [8].

$$\kappa(\boldsymbol{H}) = \frac{\sigma_1(\boldsymbol{H})}{\sigma_{r_H}(\boldsymbol{H})} \tag{14}$$

$$= \frac{\max(\{|\lambda_i^h|\}_{i=1}^{r_H})}{\min(\{|\lambda_i^h|\}_{i=1}^{r_H})} \tag{15}$$

Since the matrix of estimated taps $\hat{\boldsymbol{C}}$ approximates the inverse of $\boldsymbol{H}$, their condition numbers will be close.

MMSE EQUALIZER

The Zero-Forcing equalizer tends toward instability due to the phenomena of inverted noise described above. The MMSE Equalizer remedies this by minimizing the exepected value of the squared error.

Algorithm 1: ZF equalization

Input: data $\boldsymbol{y} \in \mathbb{R}^L$, FFT of known periodic pilot sequence $(\boldsymbol{F}\boldsymbol{x_p}) \in \mathbb{C}^{\boldsymbol{N}}$
Output: equalized data $\hat{\boldsymbol{x}}$
1 compute $\boldsymbol{F}\boldsymbol{y}$ with an FFT
2 compute $\{\hat{\lambda}_i^c\}_{i=1}^N$ with equation (11)
3 **for** $b = 1 \dots (L/N - 1)$ **do**
4 | compute $\hat{\boldsymbol{x}}_{\boldsymbol{b}}$ as in (12) and (13)
5 **end**

$$\min_{\boldsymbol{c}} E\left[\left|\boldsymbol{c}^{\boldsymbol{T}}\boldsymbol{y_n} - x[n]\right|^2\right] \tag{16}$$

with

$$\boldsymbol{y_n} = [y[n+N_2] \dots y[n-N_1]]^T \tag{17}$$

The optimal solution can be computed setting the gradient with respect to $\boldsymbol{c}$ equal to $\mathbf{0}$, then solving the system of equations. Denote the cost function as J_{MMSE}.

$$\nabla_{\boldsymbol{c}} J_{MMSE} = E\left[(\boldsymbol{c}^{\boldsymbol{T}}\boldsymbol{y_n} - x[n])\boldsymbol{y_n^H}\right] \tag{18}$$
$$= \boldsymbol{c}^{\boldsymbol{T}}\boldsymbol{R_y} - \boldsymbol{R_{xy}} \tag{19}$$
$$= 0 \tag{20}$$

which implies

$$\boldsymbol{c}^{\boldsymbol{T}}\boldsymbol{R_y} = \boldsymbol{R_{xy}} \tag{21}$$

with

$$\boldsymbol{R_y} = E[\boldsymbol{y_n}\boldsymbol{y_n^H}] \tag{22}$$
$$\boldsymbol{R_{xy}} = E[x[n]\boldsymbol{y_n^H}] \tag{23}$$

As demonstrated above, the solution depends on $\boldsymbol{R_y}$, the autocorrelation of data $\boldsymbol{y}$, as well as $\boldsymbol{R_{xy}}$, the correlation between $\boldsymbol{y}$ and $\boldsymbol{x}$. While it is true that equation (21) can be solved with matrix inversion or a pseudo-inverse, it is sometimes more efficient to find an optimal solution $\boldsymbol{c}$ iteratively. Another complication is that the distribution of $\boldsymbol{y}$ depends on the additive noise and is not always known exactly. Due to the above considerations, the solution presented here is computed with a form of stochastic gradient descent.

In particular, the least mean squares (LMS) algorithm is used. Essentially, the gradient in line (19) iteratively updates the weights $\boldsymbol{c}$. However, instead of using correlations described in (22) and (23), the raw products of the variables without taking expectation are used instead [9]. The step sizes $\boldsymbol{\mu} \in \mathbb{R}^{L_p}$ control the rate of convergence of taps while training over pilot sequence of length L_p.

$$\boldsymbol{c^{(n+1)}} = \boldsymbol{c^{(n)}} + \mu_n(x[n] - \boldsymbol{y_n^T c^{(n)}})\boldsymbol{y_n^*} \tag{24}$$

The LMS algorithm (Algorithm 2) is displayed below. For simplicity, let $\boldsymbol{y}$ be indexed from 0 to $L-1$.

Algorithm 2: MMSE equalization with LMS algorithm

Input: data $\boldsymbol{y} \in \mathbb{R}^L$, known pilot sequence $\boldsymbol{x_p} \in \mathbb{R}^{\boldsymbol{L_p}}$, step sizes $\boldsymbol{\mu} \in \mathbb{R}^{L_p}$
Output: equalized data $\hat{\boldsymbol{x}}$

initialize the weights $\boldsymbol{c^{(0)}}$ to $\mathbf{0}$
for $n = 0 \dots L_p - 1$ **do**
  update $\boldsymbol{c^{(n+1)}}$ as in equation (24) using $\boldsymbol{c^{(n)}}$, $x_p[n]$, and $\boldsymbol{y_n}$
end
for $n = L_p \dots L - 1$ **do**
  $\hat{x}[n] = (\boldsymbol{c^{(L_p)}})^T \boldsymbol{y_n}$
end

CMA EQUALIZER

The constant modulus algorithm (CMA) equalizer trains the taps by leveraging the knowledge that the transmitted data in the I/Q channels lies on a circle with fixed radius. Moreover, it finds the $\boldsymbol{c}$ that minimizes the cost function (25), then equalizes the data using such taps.

$$J_{CMA} = E\left[\left(|\boldsymbol{c^T y_n}|^2 - A^2\right)^2\right] \tag{25}$$

with

$$A^2 = \frac{E[|x[n]|^4]}{E[|x[n]|^2]} \tag{26}$$

The vector $\boldsymbol{c}$ is updated iteratively using the steepest descent algorithm

$$\boldsymbol{c^{(b+1)}} = \boldsymbol{c^{(b)}} - \mu \nabla J_{CMA}^{(b)} \tag{27}$$

The expression for $\nabla J_{CMA}^{(b)}$ is shown in equation (28). Its derivation is omitted here, but can be found in [7].

$$\nabla J_{CMA}^{(b)} = \frac{2}{N} (\boldsymbol{z^{(b)}})^T \boldsymbol{Y} \tag{28}$$

with

$$z^{(b)}[n] = (x^{(b)}[n](x^{(b)}[n])^* - A^2) x^{(b)}[n] \tag{29}$$

$$\boldsymbol{Y} = [\boldsymbol{y_{-L_1}} \cdots \boldsymbol{y_{L_2}}] \tag{30}$$

Each $\boldsymbol{y_n}$ is defined as in (17). The resulting CMA Algorithm 3 is shown below. The final output $\boldsymbol{x^{(B)}}$ contains the equalized data.

Algorithm 3: CMA equalization

Input: data $\boldsymbol{y} \in \mathbb{R}^L$, amplitude squared of transmitted data A^2, step size μ
Output: equalized data $\boldsymbol{x^{(B)}}$

initialize the weights $\boldsymbol{c^{(0)}}$ to $\mathbf{0}$
compute $\boldsymbol{x^{(0)}} = \boldsymbol{c^{(0)}} * \boldsymbol{y}$
for $b = 1 \dots B$ **do**
    update $\boldsymbol{c^{(b)}}$ as in equation (27) using $\boldsymbol{c^{(b-1)}}$ and $\boldsymbol{x^{(b-1)}}$
    compute $\boldsymbol{x^{(b)}} = \boldsymbol{c^{(b)}} * \boldsymbol{y}$
end

HARDWARE CONSIDERATIONS AND FPGA TECHNOLOGY

The novelty of this project in comparison with [5] is the implementation in FPGAs via handcoding as well high level synthesis (HLS) tools. Since FPGAs can be configured at the hardware level to perform a specific task, they benefit from lower latency more deterministic timing relative to a graphics processor unit (GPU) solutions. Furthermore, novel tools such as Vivado HLS are beginning to show potential for shortening development time of FGPA products and reducing the complexity of more advanced computations, such as floating point arithmetic. While fixed point is more efficient, it is more susceptible to overflow and underflow exceptions. Due to this, the range of possible numbers must be taken into account by the development team.

Another advantage of FPGAs is that they only use the resources required. For example, if it is known the required size of the filter is significantly less than the number of multipliers, the chip can avoid powering the unnecessary circuitry – thus saving in power consumption.

RESULTS

For the HLS implementations, fixed and floating point ZF equalizers were implemented with Vivado HLS 2017.1. A hand coded ZF was implemented in Vivado 2017.1 and the CMA and MMSE were hand coded in Vivado 2016.3. All the hand coded implementations use fixed point arithmetic. Performance of the various implementations was measured by comparing BER plots obtained without applying error correction. Error correction was omitted for simplicity since the emphasis is on relative performance. For each SNR level, 30 packets of SOQPSK data containing 512 data bits in addition to the pilot sequence were tested. To speed up simulation, trained taps were extracted from the RTL simulation and applied to the distorted data as a time domain FIR in Python 2.7.

The results for channels 1-10 from EAFB [6] are displayed in the subplots of Figure 1 . Note that these BER plots average performance over many packets. For the zero forcing examples, the bit error for different packets at the same SNR varied significantly. As an example, for the HLS fixed point ZF equalizer filtering channel 5 at 6dB SNR, the highest BER measured was 0.474 and the lowest was 0.169.

Overall, each ZF equalizer achieved roughly the same BER. The MMSE consistently outperformed the ZF implementations by as much as 15dB at low SNR, but plateaued at higher SNR. A possible reason for the plateauing is due to the gradient step overstepping the global minimum of its objective value due to the step sizes chosen. The SNR difference between the two methods is more pronounced than in [5]. A likely reason for this is since the ZF implementation only uses 32 bits of the preamble while the MMSE is trained on the entire preamble and asm pilot sequence. The small number of samples used for training the ZF equalizer were selected to guarantee the assumption of cyclic convolution for as large a channel impulse response as possible. Comparison between the MMSE and CMA conducted in [5] shows the relationship between CMA and MMSE performance. Due to time constraints, we did not repeat the simulation for CMA.

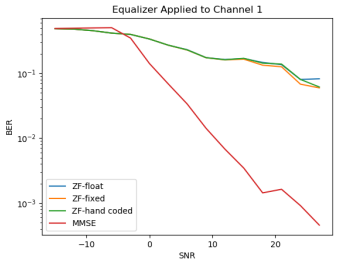
Resource utilization for each equalizer is displayed in Table 2. The CMA equalizer used the least resources. The MMSE used slightly more flip flops (FF) and look-up-tables (LUT) since it is not a blind algorithm and requires comparison of the input to a known sequence. All the ZF equalizers used significantly more FFs and LUTs than the other two. This can be explained by the complex divide operation which requires more logic to compute than other arithmetic functions. Of the two HLS ZF equalizers, the floating point used the most resources. Surprisingly, the hand coded ZF was less efficient than the HLS modules – possibly a result of the specific method and corresponding intellectual property (IP) libraries chosen for computing the complex divide. The MMSE used the most DSP slices since it includes a hand coded FIR. It is likely that this quantity would decrease if a Vivado IP library FIR implementation was used. The ZF equalizers in conrast use FFTs instead of FIRs and rely heavily on IP.

TECHNOLOGICAL MATURITY OF HLS

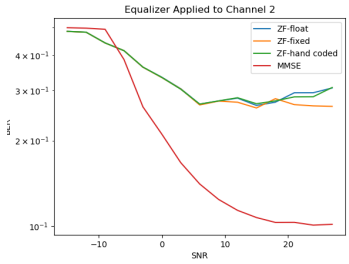
One of Vivado HLS' greatest claims is its software centric development process, enabling rapid prototyping of designs. An experienced developer can certainly achieve the advertised decrease in time to market, but the initial learning curve is complicated by a nonintuitive programming model. Specifically, while implementing our zero forcing fixed point design in software, explicit considerations for bit growth and scaling were necessary for functional correctness. The appeal of HLS' software centric development flow breaks down when hardware considerations spill into the software design process. Furthermore, learning curve was further lengthened by insufficient support for resolving errors. For example, successfully converting C++ to HDL code without error required cross checking the documentation of FFT IP, with its configuration IDE in Vivado and the header files supplied in pre-made examples. Conversely, those willing to put in the upfront effort of learning HLS stand to benefit from its accelerated development process. In particular, when coupled with available Xilinx IP, it can allow for a diverse set of applications.

CONCLUSION

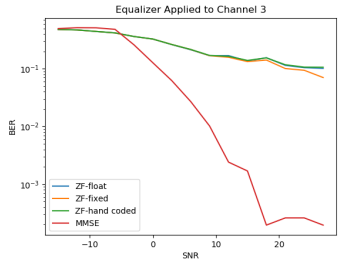
Various equalizers including ZF, MMSE, and CMA equalizers were analyzed in this paper. Each equalizer was implemented in FPGAs for comparison of resource utilization and performance



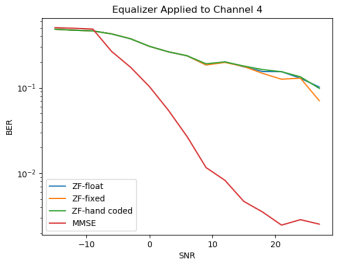
(a) Channel 1



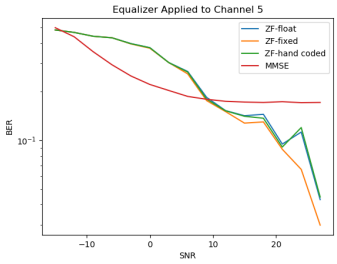
(b) Channel 2



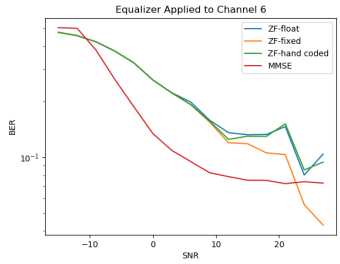
(c) Channel 3



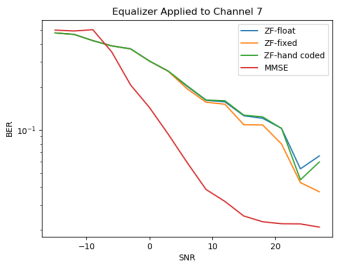
(d) Channel 4



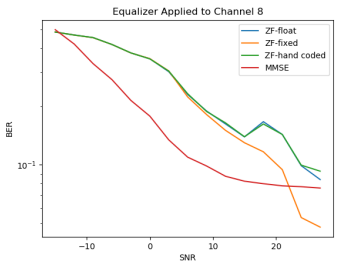
(e) Channel 5



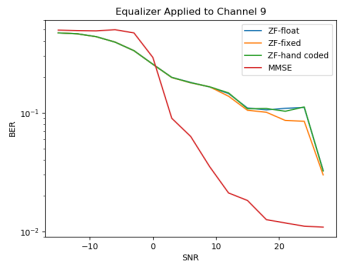
(f) Channel 6



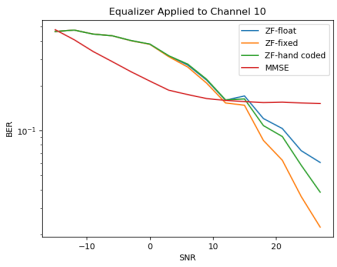
(g) Channel 7



(h) Channel 8



(i) Channel 9



(j) Channel 10

Figure 1: BER plots of equalizers for each channel

Table 2: Resource Utilization

	BRAM	DSP	FF	LUT
CMA	0	4	5589	3386
MMSE	0	226	8566	6836
Zero-Forcing (HLS - Fixed Point)	2	52	12831	10028
Zero-Forcing (HLS - Floating Point)	10	72	18599	15803
Zero-Forcing (Hand Coded)	2	42	26594	17757

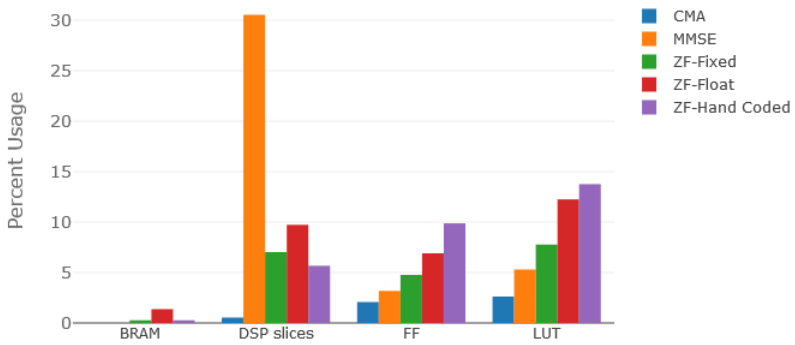


Figure 2: Resource util. normalized to available recources of the Artix-7 AC701 evaluation platform

verification. Specifically, two ZFs were implemented using Vivado HLS with fixed and floating point arithmetic. It was found that although the HLS solutions require less typing, they have an unintuitive design flow. Furthermore, the documentation and user support of HLS is not always available for troubleshooting problems. Overall, HLS can be used for rapid prototyping, but first requires an upfront effort in learning how to properly use it.

REFERENCES

[1] N. Benvenuto, R. Dinis, D. Falconer, and S. Tomasin, "Single carrier modulation with nonlinear frequency domain equalization: An idea whose time has comeagain," *Proceedings of the IEEE*, vol. 98, no. 1, pp. 69–96, 2010.

[2] M. Rice, M. Afran, M. Saquib, *et al.*, "Performance limits on joint estimation of frequency offset, channel and noise variance in aeronautical telemetry," in *International Telemetering Conference Proceedings*, International Foundation for Telemetering, 2015.

[3] "Irig standard 106-17," *Telemetry Standards*, pp. 2–4–2–14, 2017.

[4] T. J. Hill, "An enhanced, constant envelope, interoperable shaped offset qpsk (soqpsk) waveform for improved spectral efficiency," in *International Telemetering Conference Proceedings*, International Foundation for Telemetering, 2000.

[5] M. Rice, M. Saquib, M. S. Afran, A. Cole-Rhodes, and F. Moazzami, "On the performance of equalization techniques for aeronautical telemetry," in *Military Communications Conference (MILCOM), 2014 IEEE*, pp. 456–461, IEEE, 2014.

[6] M. Rice and M. Jensen, "A comparison of l-band and c-band multipath propagation at edwards afb," tech. rep., AIR FORCE FLIGHT TEST CENTER EDWARDS AFB CA, 2011.

[7] M. Rice, M. Saquib, A. Cole-Rhodes, F. Moazzami, and E. Perrins, "Final report: Preamble assisted equalization for aeronautical telemetry (paq)," 2017.

[8] B. N. Datta, *Numerical linear algebra and applications*, vol. 116. Siam, 2010.

[9] S. Haykin and B. Widrow, *Least-mean-square adaptive filters*, vol. 31. John Wiley & Sons, 2003.