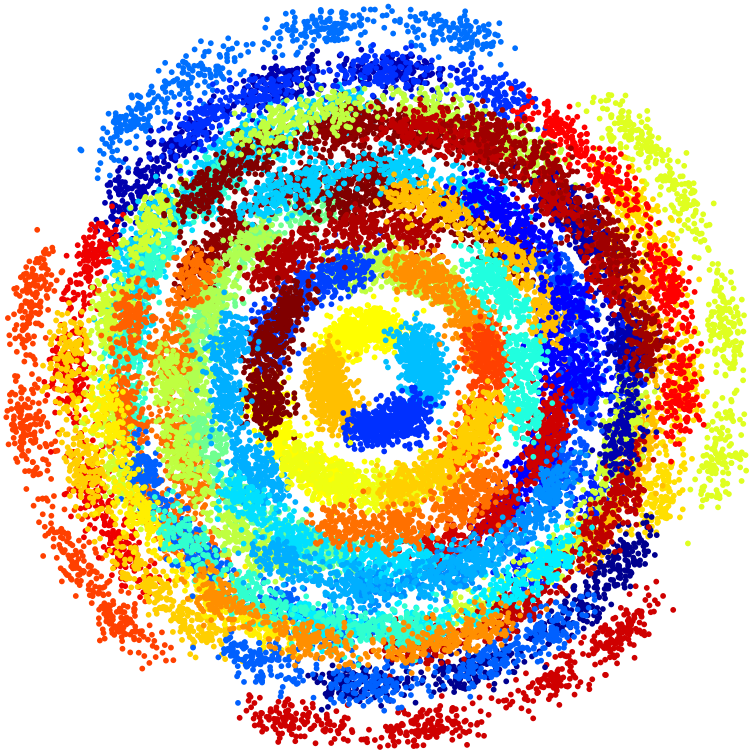


Implementation of Blind Carrier Phase Recovery for Coherent Fiber-Optical Receivers

Master's Thesis in Embedded Electronic System Design

ERIK BÖRJESON

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2018

Master's thesis 2018

Implementation of Blind Carrier Phase Recovery for Coherent Fiber-Optical Receivers

ERIK BÖRJESON

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2018

Implementation of Blind Carrier Phase Recovery for
Coherent Fiber-Optical Receivers
ERIK BÖRJESON

Supervisors:
Per Larsson-Edefors, Department of Computer Science and Engineering
Christoffer Fougstedt, Department of Computer Science and Engineering
Lars Lundberg, Department of Microtechnology and Nanoscience

Examiner:
Lena Peterson, Department of Computer Science and Engineering

Master's Thesis 2018
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Illustration of 64QAM signal with phase noise.

Typeset in LaTeX
Gothenburg, Sweden 2018

Implementation of Blind Carrier Phase Recovery for
Coherent Fiber-Optical Receivers
ERIK BÖRJESON
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

The use of higher-order modulation formats has been suggested as one way to increase the speed of fiber-optical systems. These formats are more sensitive to phase noise, which is why blind phase-search (BPS) has been suggested as one way to perform carrier recovery. In this thesis we demonstrate the feasibility of BPS by developing a hardware implementation and synthesizing it using a regular-V_t, 28 nm standard cell library. The power consumption of the design, at a symbol rate of 40 GBd, is found to be lower than 2.0 pJ/bit for 16QAM and lower than 3.1 pJ/bit for 64QAM. Higher symbol rates can also be used, without significantly affecting the energy/bit. Fixed-point aspects of the design are examined and we show that using 7 bits for 16QAM and 8 bits for 64QAM result in a penalty of less than 0.3 dB, compared to floating point. We also present a hardware-efficient method to approximately calculate the distance to the closest constellation point, without significantly affecting the bit-error rate (BER). Furthermore, the feasibility of joint processing of multiple channels sharing the same phase noise is evaluated and phase estimation is shown to consume approximately 5 times the power of the phase compensation.

Keywords: fiber-optical communication, phase noise, carrier recovery, blind phase-search, digital signal processing, power consumption

Acknowledgements

I would like to thank my supervisors, Prof. Per Larsson-Edefors, for invaluable feedback and discussions; Christoffer Fougstedt, for much appreciated help with the software tools used throughout the project; and Lars Lundberg, for suggesting the thesis topic and providing valuable insights into the theoretical aspects of fiber-optic communication systems. Finally, I would like to thank Lena Peterson for helpful comments on both the languange and content of this thesis.

Erik Börjeson, Gothenburg, June 2018

Contents

Acronyms

ADC	Analog-to-digital converter
ASIC	Application-specific integrated circuit
ASK	Amplitude-shift keying
BER	Bit-error rate
BPS	Blind phase-search
CD	Chromatic dispersion
DCF	Dispersion-compensating fiber
DFF	Dispersion-flattened fiber
DFT	Discrete Fourier transform
DSF	Dispersion-shifted fiber
DSP	Digital signal processing
FEC	Forward error correction
FIR	Finite impulse-response
IF	Intermediate frequency
LO	Local oscillator
LUT	Lookup table
MCM	Multiple constant multiplication
MIMO	Multiple-input and multiple-output
OOK	On-off keying
PBC	Polarization beam combiner
PBS	Polarization beam splitter
PLL	Phase-locked loop
PMD	Polarization-mode dispersion
PSK	Phase-shift keying
QAM	Quadrature amplitude modulation
QPSK	Quadrature phase-shift keying
SAIF	Switching activity interchange file
SNR	Signal-to-noise ratio

1

Introduction

Fiber-optical communication systems are the most common communication links for high-volume data transfer today, especially for medium to long range transmissions. These links make up the backbone of the Internet and are widely used in the digital infrastructure. An ideal fiber-optical communication system has high data rate, can cover a long distance while avoiding signal regeneration, incurs no errors to the transmitted data, and has a low power consumption. Unfortunately, the fibers and other components available are not ideal, as noise and other artifacts are added to the signal when it is transmitted over the system. Additionally the spectral and power efficiency is limited by the system components.

Network data-throughput demand is expected to increase with time [1], which means that networks supporting higher data rates without a significant increase in power consumption are needed. A high power consumption is not just directly related to cost for the network operator, but also to environmental impacts and heat dissipation problems of the equipment used.

One way of enabling higher transmission rates is by using higher-order modulation formats. As the complexity of the modulation increases, the requirements on the optical receiver's digital signal processing (DSP) unit, used to decode the data, get stricter. One of the functions of the receiver is to compensate for signal impairments caused by the nonidealities of the transmission fibers and the lasers generating the carrier and local oscillator (LO) waves. The phase noise of the lasers constitutes one such impairment and higher-order modulation schemes are more sensitive to this type of error. This sensitivity makes the phase-noise compensation unit a demanding unit in the DSP architecture.

If several channels share the same phase noise, i.e. use the same lasers to generate the carrier, and are decoded in the same transceiver, it is possible to implement joint processing of the signals. Reasons for using a joint processing implementation include power and area savings, and one proposed way to implement such a system is to use a master-slave design. In this type of design, the results from phase-noise calculations on one channel are reused for all other channels. Such a solution has been demonstrated by Lundberg et al. [2], for a system transmitting data over comb-based superchannels and using a blind phase-search (BPS) algorithm.

1.1 Aim

To the best of our knowledge, no publications on application-specific integrated circuit (ASIC) implementations of BPS phase recovery exist in the open literature and, thus, no power-consumption data are available. Therefore, the aim of this thesis is to develop such an implementation and use it to evaluate the potential of BPS for carrier recovery in a fiber-optical system. An ASIC design enables the retrieval of power-consumption data, silicon-area demand and can be used to evaluate the effect of performing BPS carrier recovery in a fixed-point environment. Aspects of the master-slave approach can also be evaluated, such as the potential energy savings. The aim of this thesis is to provide insights into the feasibility of using BPS and the master-slave approach for carrier recovery in future fiber-optical communication systems.

1.2 Thesis Outline

The outline of this thesis mirrors the work flow used during the thesis project and starts with a short background of fiber-optical communication systems followed by a description of BPS carrier recovery, in Chapter 2. The following chapter presents the details of the hardware implementation and its components. In Chapter 4, the implemented design is evaluated and different parameter settings are studied. The thesis ends with a conclusion in Chapter 5.

2
Background

This chapter starts with a brief introduction to fiber-optical communication systems and modulation formats, focused on the coherent receivers used in this thesis. The different parts of a receiver DSP unit typically used in current systems are discussed and finally the details of the BPS algorithm are described.

2.1 Fiber-Optical Communication Systems

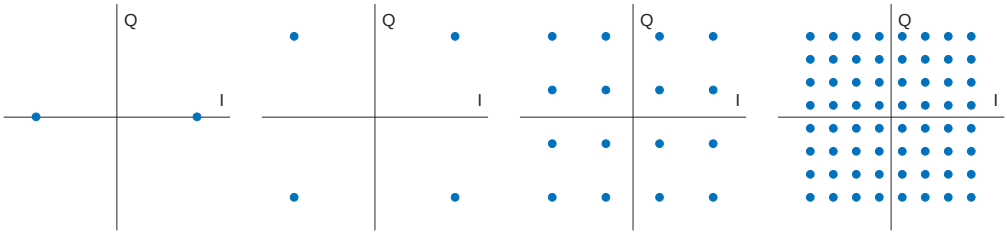
A generic fiber-optical communication system can be divided into three main parts, a transmitter, a communication medium and a receiver. In a fiber-optical system, the medium is an optical fiber which is usually made of a silica-glass or plastic core and a cladding. The fiber is designed using cladding with a lower refractive index than the core, which ensures that a ray with an incident angle less than a critical angle is confined to the fiber [3]. The transmitter converts an digital bit stream into an analog optical signal and launches it into the fiber. The role of the receiver is to convert the transmitted signal back into an electrical one.

When the optical signal is generated and transmitted over the fiber, the signal integrity is degraded as noise and other artifacts are added to the signal. These distortions need to be accounted for by the receiver in order to recover the input signal correctly. If a digital signal is used, the bit-error rate (BER) can be used as a measurement on how well the system performs in terms of signal recovery. The BER is defined as the average probability of incorrectly decoding a bit [3]. For a system to be considered error-free, the BER should typically be below 10^{-15} [4].

2.1.1 Modulation Formats

Data sent over a fiber-optical communication link can be encoded using different modulation formats, the simplest one being on-off keying (OOK). This format is a type of amplitude-shift keying (ASK) where the amplitude of the signal is modulated to represent the data. In OOK, shown in Fig. 2.1a, the optical signal is either off or on, representing a zero or a one.

The phase of the optical signal can also be used to carry data using phase-shift keying (PSK). When using ASK or PSK the amplitude or phase can take on more than two values, which gives the possibility to code more than one data bit per transmitted symbol, e.g. quadrature phase-shift keying (QPSK) shown in Fig. 2.1b.



(a) OOK **(b)** QPSK **(c)** 16QAM **(d)** 64QAM

Figure 2.1: Constellation diagrams for four common modulation formats.

The number of bits can be increased further by combining the two modulation formats and simultaneously code data on both the amplitude and phase. Examples of this type of coding are the quadrature amplitude modulation (QAM) formats, such as 16QAM and 64QAM shown in Fig. 2.1c and Fig. 2.1d, respectively. Since each symbol can be seen as a complex number, a rectangular representation is often used in the electrical domain. In this representation the amplitude and phase of a symbol is represented by its real, or in-phase, part (I) and the imaginary, or quadrature, part (Q).

An additional doubling of the data rate can be achieved by transmitting two orthogonal polarizations simultaneously [3]. In order to use the higher-order modulation formats, such as 16 and 64QAM, coherent detection methods are needed [5], which preserve both the amplitude and the phase of the incoming signal. These methods are presented in more detail in Section 2.1.3.

The higher-order formats allow for more data to be transmitted per symbol, but as the distance between symbols decreases, the system's sensitivity to distortion increases [3]. To compensate for this sensitivity, the system might need higher quality components and additional processing is likely needed on the receiver side to distinguish between the symbols.

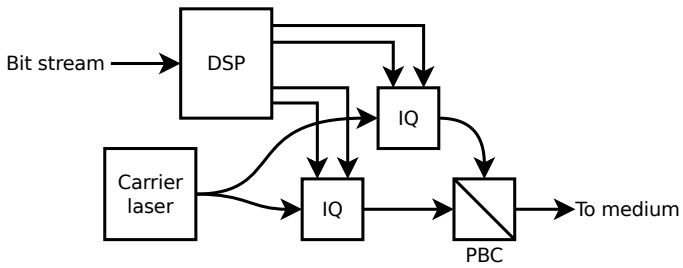


Figure 2.2: Schematic of a fiber-optical transmitter.

2.1.2 Transmitter

The transmitter, shown in Fig. 2.2, converts a digital electrical bit stream into an optical signal by modulating a carrier wave. In the coherent transmission systems described in this thesis, a laser is typically used as the source of this carrier [3]. A critical property of the laser is the linewidth, which is a measure of the laser output

spectrum width and related to laser phase noise. The laser linewidth is inversely proportional to the coherence time, a time in which the phase is relatively stable [3].

In coherent transmission systems the bit stream to be sent is first encoded in a transmitter DSP, using the selected modulation format, and the resulting electrical signals are used to modulate the carrier wave in an IQ modulator [6]. This modulator controls both the amplitude and phase of the outgoing signal to represent the symbols in the modulation format. Output signals from two modulators can be combined in a polarization beam combiner (PBC) to send data over two polarizations [3].

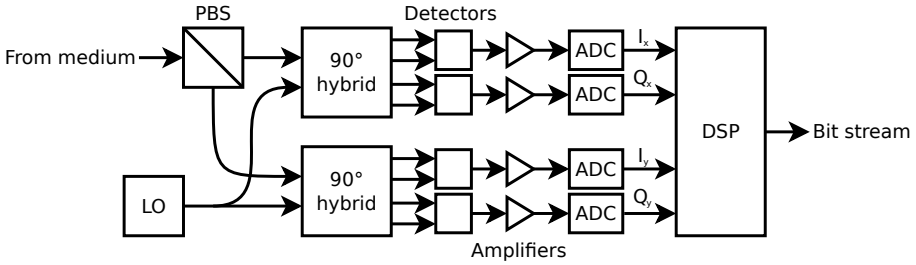


Figure 2.3: Schematic of a coherent receiver.

2.1.3 Receiver

This thesis considers only coherent detection techniques, since these are needed to decode the higher-order modulation formats used. The architecture of a coherent receiver is shown in Fig. 2.3 and the first stage is splitting the input signal into two polarizations using a polarization beam splitter (PBS), after which the two polarizations are mixed separately with a local oscillator (LO) laser in two 90° optical hybrids [7]. There are three main coherent receiver architectures: homodyne, intradyne and heterodyne, which differ in the LO frequency used. In the case of homodyne detection the LO frequency is selected to be the same as the carrier frequency, which ideally results in a zero intermediate frequency (IF) [3]. If the LO frequency is selected to be very close to the carrier frequency, intradyne mode is used, which results in a non-zero IF. The advantage is that the need to carefully control the frequency and phase of the LO laser, e.g. using a phase-locked loop (PLL), is reduced [3]. The remaining IF can be handled in DSP.

Heterodyne detection, i.e. when the LO frequency is different from the carrier, can also be used, but has a lower signal-to-noise ratio (SNR) improvement than the homodyne and intradyne versions [3]. The higher IF used in heterodyne receivers also means that detectors and analog-to-digital converters (ADCs) with higher bandwidth is needed.

The 90° hybrids output four signals each, which represent the in-phase (I) and quadrature (Q) parts of the signal as well as their inverses. The signals are fed to balanced photodetectors that convert the optical signals to electrical ones [7]. The result is a conversion from an analog optical signal, with amplitude and phase, to two electrical signals that can be described as a complex number, $Z = I + iQ$ [3]. Converting the signal to this form makes it possible to represent both the amplitude

and phase of the optical signal using only two electrical amplitudes. To be able to further digitally process the received signal, the analog electrical signals are amplified and digitized using ADCs.

2.2 Digital Signal Processing

The signal flow of the DSP in a coherent optical receiver is shown in Fig. 2.4 and consists of a number of subsystems [7]. These subsystems are used to compensate for signal impairments caused by the non-idealities of the optical fiber and other components.

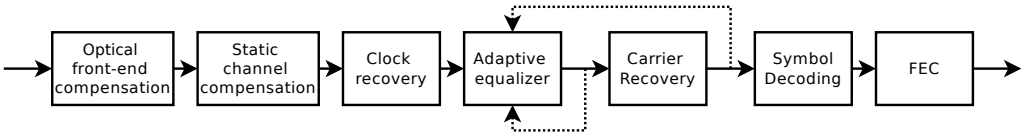


Figure 2.4: Block diagram showing the DSP chain of a typical coherent receiver.

2.2.1 Optical Front-End Compensation

The optical front-end of the receiver is not perfect and can distort the input signal. If the four optical paths of the receiver are of different lengths or the response of the photodiodes or amplifiers is uneven, timing mismatches will be introduced, which will make the electrical signals unsynchronized [7]. With the high data rates used, the skew can be several samples in length. By compensating for the skew in DSP, the requirements on the optical components can be reduced, while still maintaining a low BER [8]. An imperfect 90° hybrid will also distort the optical signal by making the angle between I and Q differ from the 90° optimum. This is also compensated for in DSP, using orthogonalization algorithms [7].

2.2.2 Static Channel Compensation

The power of a light pulse that is transmitted over a fiber-optic system is spread over a wider time interval as it travels down the fiber, so called chromatic dispersion (CD) [9]. This effect can cause adjacent symbols to interfere with each other, making correct symbol detection difficult. To compensate for CD, methods such as using dispersion-shifted fibers (DSFs), dispersion-flattened fibers (DFFs) or dispersion-compensating fibers (DCFs) can be used [9]. The compensation can also be carried out in DSP using CD compensating filters. Since CD is a static impairment, dependent on fiber properties and length, the compensation can be implemented as a finite impulse-response (FIR) filter with static coefficients [10].

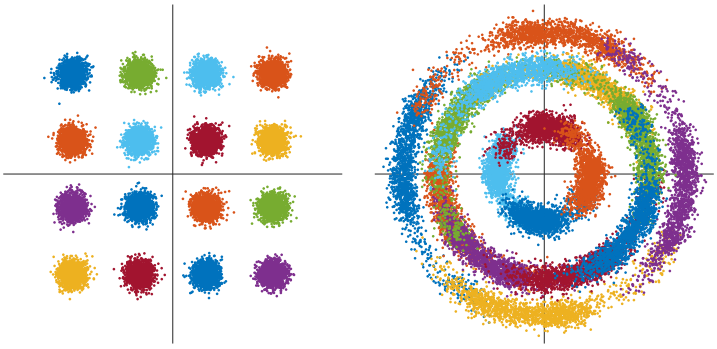
2.2.3 Clock Recovery

Once the DSP has compensated for the static impairments of the optical system, a synchronization between the transmitter's symbol clock and the ADC sample clock is performed. Typically, the processing preceding clock recovery is carried out at the ADC sampling rate, while the subsequent processing is performed at a symbol rate derived from the received data [7]. There is a number of algorithms that can be used for this synchronization, such as using a discrete Fourier transform (DFT) to detect the clock rate or by applying an adaptive resampling FIR filter [11]. The clock recovery can also be integrated in the dynamic equalizer in the following stage [7].

2.2.4 Dynamic Channel Compensation

Dynamic channel compensation is also called adaptive equalization and is used to compensate for the time-varying impairments of the transmission system. The most salient of these impairments is polarization-mode dispersion (PMD) [7], which is caused by varying birefringence along the length of the optical fiber [3]. These variations are generated by the non-circular cross section of the fiber, small variations of the refractive index and changing environmental conditions [9]. Because of the varying birefringence, the polarizations will have different group-delays and for coherent systems the result is crosstalk between the two polarizations.

The dynamic equalizers used to compensate for PMD are typically constructed as multiple-input and multiple-output (MIMO) filters with dynamically updated taps [7]. These equalizers do not only compensate for PMD, but have the added advantage of reducing remaining static errors and improving noise rejection [12]. The error signal used to update the taps is taken either directly at the output of the equalizer or, in the case of decision-directed update, from the output of the carrier recovery stage. The latter has the added advantage of making the equalizer able to track a residual phase drift.



(a) **(b)**

Figure 2.5: Illustration of how phase noise affects a 16QAM signal, where (a) is the original signal and (b) is the same signal with added phase noise.

2.2.5 Carrier Recovery

The last stage before decoding the input signal is carrier recovery. In a coherent receiver this recovery is performed in two stages, frequency recovery and phase estimation, where the first removes the offset between the carrier and LO frequency. This offset can be removed by finding the spectrum peak of the equalized signal [7].

Phase noise originates from the linewidth of the carrier and LO lasers [3] and its effect is shown in Fig. 2.5. As the phases of the carrier and LO varies, the signal is rotated in the IQ representation used inside the coherent receiver. In order for the symbols to be detected correctly, the original signal must be recovered by removing the phase noise. The phase noise of the k-th sample, θ_k, can be modelled as a Wiener process [13]

$$\theta_k = \theta_{k-1} + \Delta\theta, \tag{2.1}$$

where $\Delta\theta$ is a random Gaussian variable with zero mean and a variance described by

$$\sigma^2_{\Delta\theta} = 2\pi\Delta f T_s, \tag{2.2}$$

where Δf is the combined linewidth of the carrier and LO lasers and T_s is the symbol period.

There is a number of different phase-estimation techniques and the simplest ones revolve around using pilot tones. These are unmodulated signals transmitted in parallel with the modulated, data-carrying signals [14]. The main problem with this approach is that the unmodulated pilot tones consume bandwidth which could have been used for carrying data.

The phase estimation can also be realized in DSP without using a pilot tone. Such algorithms are called blind, since the phase of the lasers is unknown. For simple PSK modulation formats, such as QPSK, the Viterbi-Viterbi algorithm can be used [15]. For QPSK, this algorithm removes the modulation by taking the fourth power of the signal and the phase is estimated from the resulting carrier. This method does not work well with higher modulation formats, such as QAM, since these contain symbols of different amplitudes. An alternative approach is to use BPS, which is described in Section 2.3.

2.2.6 Forward Error Correction

As previously mentioned, the BER of a modern fiber-optical system is typically required to be $< 10^{-15}$ [4]. To achieve such a low error rate, the system often relies on forward error correction (FEC) to reduce the requirements on others parts of the system. By letting the transmitter code the signal in a redundant way, the receiver can decode the signal and recover from a limited number of errors. The cost is an added overhead of the extra redundant data transmitted. The FEC is typically inserted as the last block in DSP chain and state of the art FECs can handle a pre-FEC BER of over 10^{-2}, while maintaining a post-FEC BER of 10^{-15} [4].

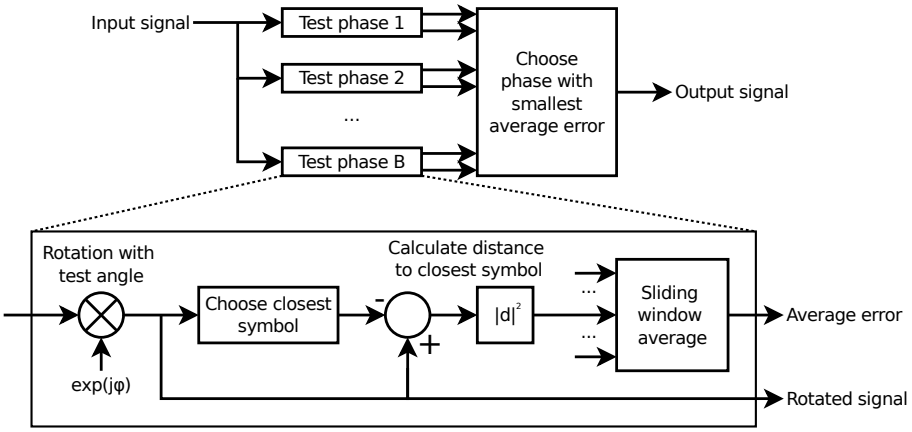


Figure 2.6: Schematic of the BPS algorithm.

2.3 Blind Phase-Search

BPS was first suggested for use in fiber-optical systems in [13] and a schematic of its structure is shown in Fig. 2.6. The algorithm is applied on the equalized and downsampled signal, where one sample represents one symbol. Each input symbol (Z_k) is rotated with B number of test phases (φ_b), spread over an angle of $\pi/2$ calculated as [13]

$$\varphi_b = \frac{b}{B} \cdot \frac{\pi}{2}, b \in \{0, 1, ...B-1\}. \tag{2.3}$$

An illustration of symbol rotation in the complex plane is shown in Fig. 2.7. The symmetry of the QAM-symbol positions makes rotating the input sample to cover a complete circle unnecessary.

Each of the rotated symbols is compared to the constellation points in the modulation format used, and the distance between the rotated symbol and the closest point is calculated as

$$|d_{k,b}|^2 = |Z_k e^{-j\varphi_b} - \hat{X}_{k,b}|^2, \tag{2.4}$$

where $\hat{X}_{k,b}$ is the closest symbol [13]. To reduce the impact of noise on the result, a sliding average window

$$e_{k,b} = \sum_{n=-N}^{N} \frac{|d_{k-n,b}|^2}{2N}, \tag{2.5}$$

is applied to the resulting distances giving an average error for each test phase. The length of the sliding average (N) is chosen depending on the laser's linewidth and the symbol rate. The phase angle can be determined by finding the minimum $e_{k,b}$ and used to choose the correct rotated input sample to be output.

2.3.1 Reducing the Number of Test Phases

The accuracy of BPS depends heavily on the number of test phases used and the number of phases needed increases with higher modulation formats. As a larger

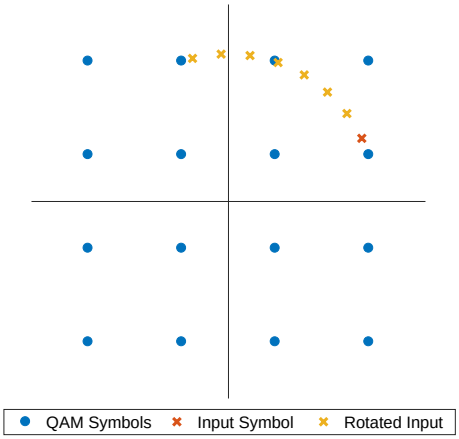


Figure 2.7: BPS algorithm illustrated for 16QAM with 8 test phase angles.

number of test phases increases the complexity of a hardware implementation significantly, Sun et al. [16] propose a method of quadratic approximation to reduce the number of phases by using interpolation. First the differences between the minimum average error $e_{k,b}$ and the errors of the two neighboring phase angles are calculated as

$$d_{k,-1} = e_{k,b-1} - e_{k,b}, \tag{2.6a}$$
$$d_{k,1} = e_{k,b+1} - e_{k,b}. \tag{2.6b}$$

These differences are used to calculate an interpolated phase angle using

$$\varphi_k = \varphi_{k,b} + \frac{\Delta\varphi}{2} \cdot \frac{d_{k,-1} + d_{k,+1}}{d_{k,-1} - d_{k,+1}}, \tag{2.7}$$

where $\Delta\varphi$ is the distance between two neighboring test phases. The process is illustrated in Fig. 2.8.

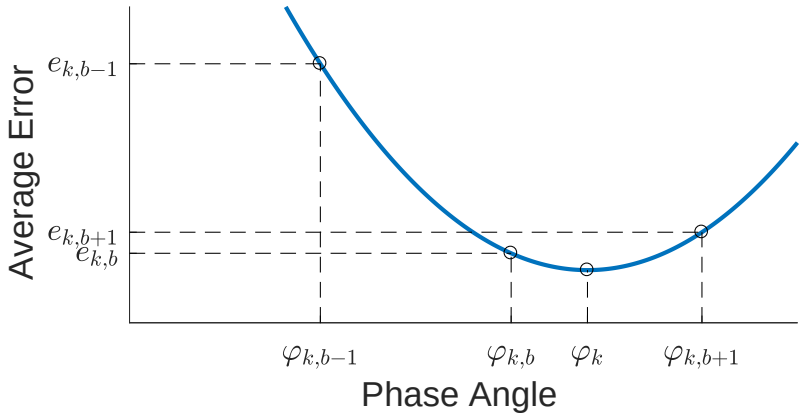


Figure 2.8: Illustration of how interpolation can be used to estimate a more exact carrier phase (φ_k) by using the phase angle ($\varphi_{k,b}$) that results in the minimum average error ($e_{k,b}$) and its two closest neighbors.

2.3.2 Joint Processing in a Master-Slave Configuration

Joint processing for carrier-phase recovery of QPSK has been proposed by e.g. Liu et al. [17], who use the Viterbi-Viterbi algorithm, and has been demonstrated for multicore [18] and multimode [19] systems. The common idea is to exploit the fact that signals originating from the same source, e.g. an optical frequency comb, share the same phase noise. If the same LO source is used for all channels on the receiver side, the carrier phase calculations from one channel can be reused for all other channels. This joint processing has the potential to reduce the resources needed for phase recovery in the DSP ASIC. Lundberg et al. [2] describe how joint processing can be applied to higher-order modulation formats using BPS.

3

Implementation

The following sections describe the implementation of the DSP units created to realize a hardware implementation of BPS, and the different approximations and simplifications used to create a feasible design. The main focus has been to create a design that consumes a minimum amount of power, while still providing a satisfying functionality and having as small a silicon footprint as possible. First, an overview of the design is presented followed by more detailed descriptions of each component.

3.1 Design Overview

An implementation of BPS needs to process data at high speeds and in order for the ASIC design to reach these speeds both pipelining and parallelization are used in the design. Pipelining is a method used to reduce the critical path of the design by inserting registers at suitable positions [20]. This reduction can be used to e.g. increase the clock speed or to reduce the power consumption. The latency of the design is, however, increased by pipelining. By parallelizing the design, i.e. performing calculations on multiple samples simultaneously, the symbol rate can be increased without increasing the clock speed. The speed gained from parallel processing can also be traded for a reduced power consumption [20].

The bulk of the design was implemented in VHDL, with parts generated in Verilog. The DSP design is divided into two main parts, an estimation unit and a unit used for compensation. The estimation unit calculates the carrier phase offset and sends an index, representing a phase angle, to the compensation unit, which carries out the actual phase compensation of the input data. This architecture simplifies implementation of a master-slave system, where one estimation unit provides phase data to many compensation units. A block diagram illustrating this setup is shown in Fig. 3.1.

3.2 DSP Units

An overview of the design of the estimation unit is shown in Fig. 3.2 and consists of subunits that perform the calculations. The input is composed of P subsequent input symbols, represented by its in-phase and quadrature (IQ) components. These symbols are processed in P parallel tracks, that first map the input symbols to the

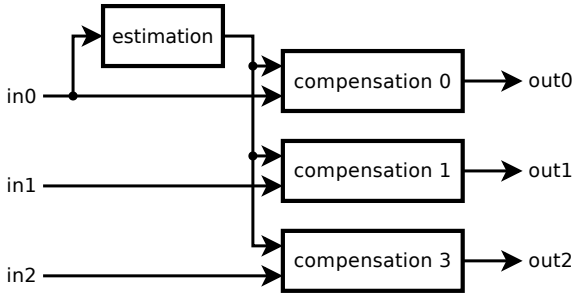


Figure 3.1: Block diagram of the DSP design with one estimation unit driving three compensation units.

first quadrant and then rotate them with B test phases. Calculation of the distance to the closest symbol is performed for each test angle in each parallel track, resulting in $B \cdot P$ distances.

For the next stage in the processing, the parallelism is abandoned and the block average of the distances for each test phase over the P parallel tracks is calculated. To be able to calculate the averages over blocks longer than P, the outer average DSP unit is inserted. The angle with the minimum average distance is found in the next processing unit, followed by interpolation of the test phases and unwrapping. Details of the design of the estimation subunits are presented in the following sections, followed by a description of the compensation unit in Section 3.2.8.

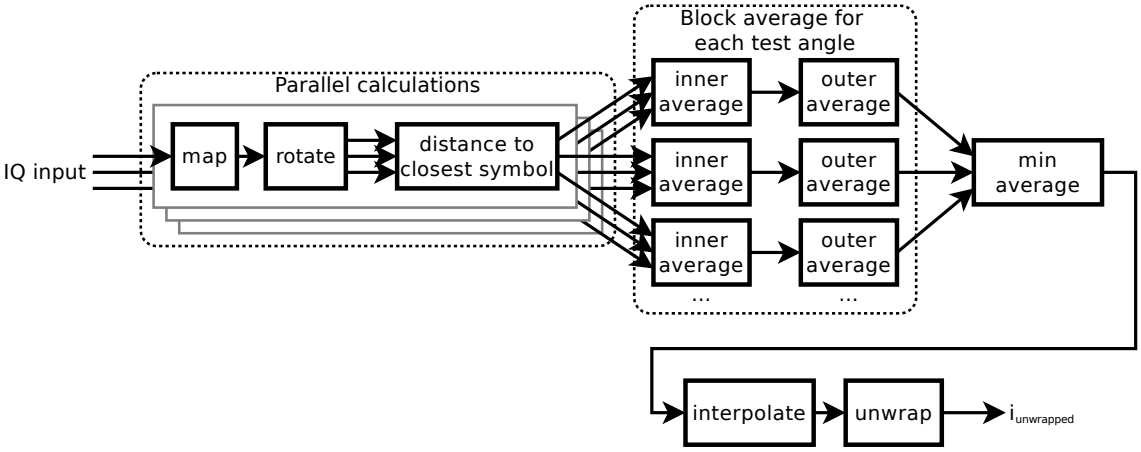


Figure 3.2: Simplified overview of the estimation DSP module.

3.2.1 Symbol Mapping

To minimize the number of bits needed for the DSP calculations, the input data can be mapped to the first quadrant. The symmetry of the symbols in a quadrature constellation makes it possible to remove the sign bit, while still maintaining the same distance to the closest symbol. An illustration of this symmetry is shown in Fig. 3.3. The mapping can be performed by rotating the input $-\pi/2$ for symbols in the second quadrant, π for the third and $\pi/2$ for the fourth. A rotation by $-\pi/2$ is equivalent to multiplying the input symbol with $e^{-j\pi/2} = 0 - 1j$, which can be

expressed as

$$(a + jb) \cdot (0 - 1j) = b - ja, \tag{3.1}$$

where $a+jb$ is the input symbol. Carrying out the same calculations for all quadrants result in the expressions shown in Table 3.1. These expressions can be further simplified to $|a| + j|b|$ for the first and third quadrants and $|b| + j|a|$ for the second and fourth.

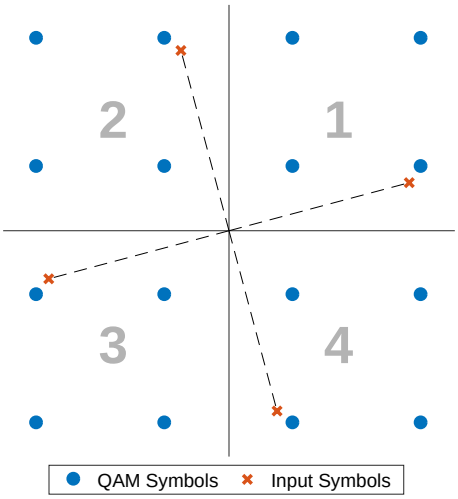


Figure 3.3: Illustration of how mapping to the first quadrant affects the distance to the closest symbol.

Table 3.1: Expressions for calculating the mapping to the first quadrant.

Quadrant	Expression
1	$a + jb$
2	$b - ja$
3	$-a - jb$
4	$-b + ja$

A block diagram of the mapping subunit layout is shown in Fig. 3.4. The absolute value of the the signed input vectors are taken and the sign bits of the original vectors are XORed to determine if the real and imaginary parts should be swapped or not. The resulting output can be made one bit shorter than the input, since the output is strictly positive.

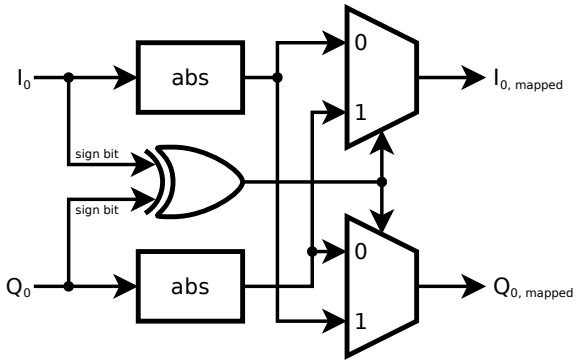


Figure 3.4: Block diagram of the mapping of symbols to the first quadrant.

3.2.2 Rotation with Test Phases

Rotations of the input symbols with a number of test phases can be performed as multiplications with complex constants, where each test phase (φ) is converted to a complex number in polar form; $r = r_I + jr_Q = \sin\varphi + j\cos\varphi$. The result of rotating the input with the first test phase can be calculated as

$$\begin{aligned} A_{\angle 0} &= A_{\text{mapped}} \cdot r_{\angle 0} = (I + jQ) \cdot (r_{\angle 0,I} + jr_{\angle 0,Q}) \\ &= (Ir_{\angle 0,I} - Qr_{\angle 0,Q}) + j(Ir_{\angle 0,Q} + Qr_{\angle 0,I}), \end{aligned} \tag{3.2}$$

where $A_{\text{mapped}} = I + jQ$ is the mapped input, and $A_{\angle 0}$ is the result of a rotation with the phase represented by $r_{\angle 0}$. From (3.2) it is clear that a rotation implemented as a complex multiplication needs four hardware multipliers and two adders. Multiplication is a costly operation in terms of power and area and these operations are performed for B number of test phases over P parallel tracks, resulting in a total of $4 \cdot P \cdot B$ multipliers. If not handled carefully, these multipliers would have a large impact on the performance of the design.

To reduce the performance impact of the multipliers, multiplierless multiple constant multiplication (MCM) is used in the design of the rotation DSP component. The Verilog modules describing the MCM components are generated using SPIRAL software and the Hcub algorithm [21]. By reducing the multiplications to a series of shifts and additions/subtractions and sharing the intermediate results over many constants, MCM decreases the hardware usage of the design. The symmetry of the complex representation of the test phases makes it possible to use two MCM blocks, instead of four, to calculate the products.

The fact that the mapped input symbols are always situated in the first quadrant and that the test phases have a maximum value of $\pi/2$, limits the multiplication product to the first two quadrants. To keep the number of bits to a minimum, results positioned in the second quadrant are rotated back to the first using a method similar to the one previously described in Section 3.2.1. This mapping removes the need for sign bits in the output from the rotation component. A block diagram of the complete rotation unit is shown in Fig. 3.5.

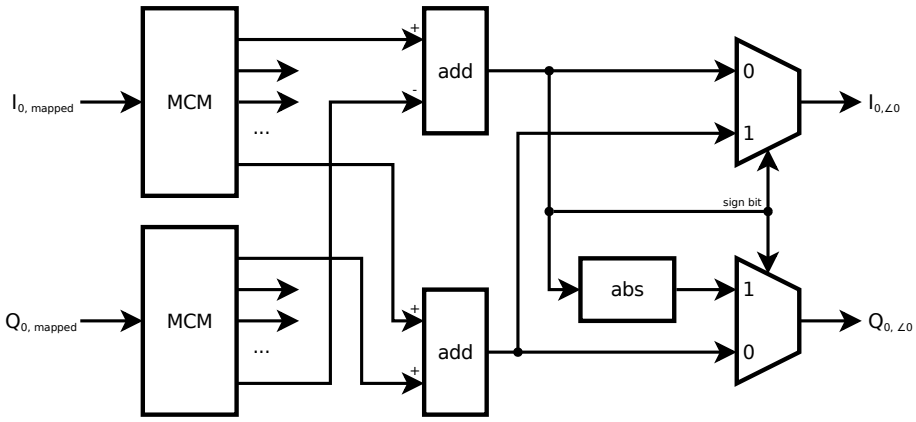


Figure 3.5: Rotation of the mapped input symbols with test angles. The structure is shown for one of the test phases.

3.2.3 Distance to Closest Constellation Point

To find the distance from the rotated input symbols to the closest constellation point, the DSP unit shown in Fig. 3.6 is used. First the row and column of each of the rotated symbols are determined using constant limits calculated for each modulation format. These coordinates are used to retrieve the constellation point from a lookup table (LUT) and the difference between the rotated symbol and the constellation point is calculated. The mapping of input symbols to the first quadrant reduces the complexity of this DSP unit significantly, as only a quarter of the constellation points need to be accounted for. Such a reduction of resource usage in this part of the design makes a large impact, as the calculations are performed for $P \cdot B$ symbols in parallel.

The floating-point BPS implementation, described in Section 2.3, uses the magnitude squared of the distances for the average calculated in the next DSP stage. This method is, however, not feasible in the fixed-point implementation considered here for two main reasons; firstly, it involves multiplications and secondly, twice as many bits are needed to hold the squared magnitude compared to the magnitude. To avoid these hardware demanding calculations the approximation

$$|a + jb| = \sqrt{a^2 + b^2} \approx \max(a, b) + \frac{\min(a, b)}{2}, \tag{3.3}$$

is used instead [22]. This approximation is implemented using only comparators, a shift and an adder, shown in the right portion of Fig. 3.6. The method is, however, only an approximation and incurs an error that is dependent on the angle of the complex input [22].

The magnitude of the difference between the rotated input symbol and the closest constellation point is limited to a smaller range of values than the input. This smaller range means that fewer bits can be used to represent the magnitude, which will make the following blocks more area and power efficient. For very large differences the design saturates the magnitude, at the maximum value defined by the word length, in order to further reduce the number of bits.

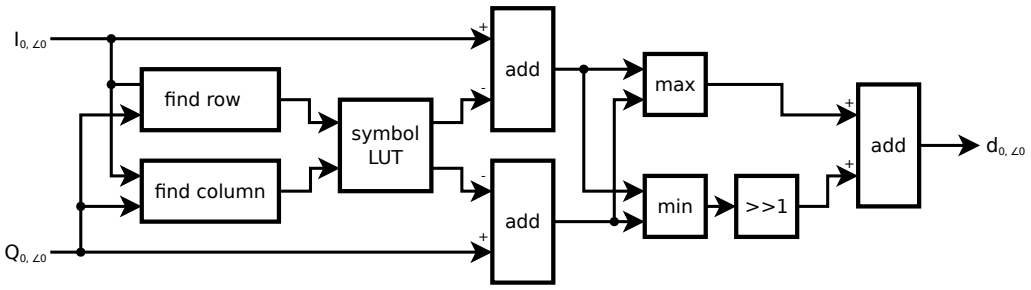


Figure 3.6: DSP block used to find the closest constellation point and calculate the distance to the input symbol.

3.2.4 Average Distance Calculation

The original BPS algorithm, presented in Section 2.3, uses a sliding window average [13]. The parallel implementation of such a calculation is, however, very hardware intensive. Instead the average distance for each test phase is calculated as a block average over all P parallel tracks, by adding all distances together and dividing by P. A block diagram of the design is shown in Fig. 3.7. The division is implemented as a right shift, which puts a limitation on the parallelization factor to powers of 2. If the average needs to be calculated for a larger block, the inner average unit is followed by the outer average, shown in Fig. 3.8. The latter stores the previous values and calculates the average over N succeeding blocks and updates the output every Nth clock cycle.

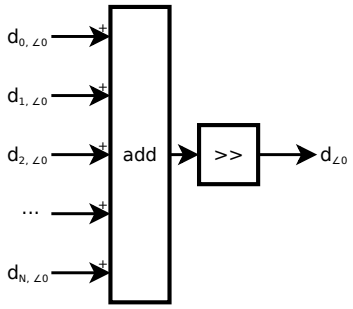


Figure 3.7: Block diagram of the inner average calculation.

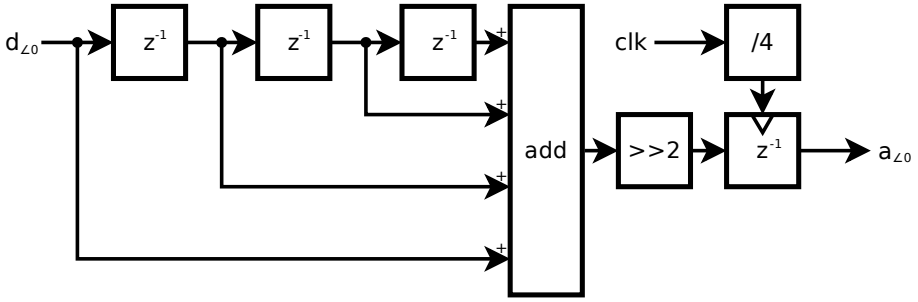


Figure 3.8: Block diagram of the outer average DSP block for lengthening the block average by a factor of 4.

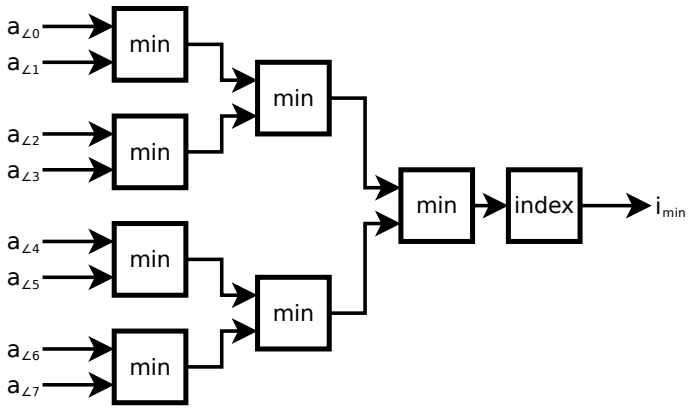


Figure 3.9: Block diagram of the comparator tree used to find the minimum average, for a design with eight test phases.

3.2.5 Minimum Average Distance

Finding of the minimum average distance is done using the `CW_minmax` component from the Cadence ChipWare library [23]. The result is a comparator tree, as shown in Fig. 3.9. The output from the tree consists of the index of the test phase resulting in the minimum value.

Other more advanced and faster ways of finding the minimum value, such as those presented in [24], have been considered. But all of these have a larger power consumption than the traditional binary tree for the number of inputs used here. For higher-order modulation formats than 64QAM, it is possible that these methods might yield better results.

3.2.6 Interpolation

The purpose of the interpolation unit is to facilitate a reduction of the number of test phases. The interpolation is performed by using a modified version of the quadratic-approximation technique described in Section 2.3.1. As the DSP calculations are performed in fixed-point arithmetic, some changes to the calculations are necessary, both because of fixed-point limitations and for performance reasons.

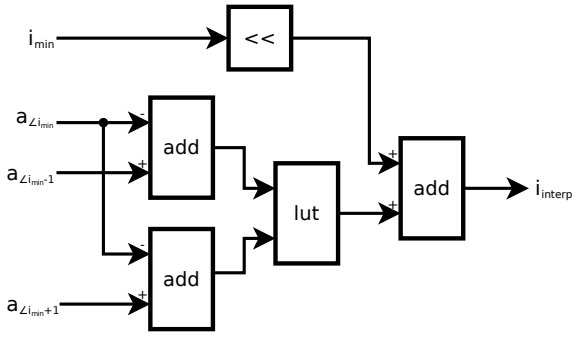


Figure 3.10: DSP unit used for interpolation.

The index of the phase resulting in the lowest average ($i_{\min}$), calculated in the previous DSP unit, is used to select the value of the minimum average distance ($a_{\angle i_{\min}}$) and average distance for the two neighboring angles ($a_{\angle i_{\min}-1}$ and $a_{\angle i_{\min}+1}$). The difference between the minimum distance and its two neighbors is calculated and the results are used to index a position in a LUT. If twice as many angles are desired, the original index is shifted one step to the left, and the value from the LUT is added to it. The LUT values are calculated, based on the number of test phases and the number of intermediate steps wanted, to either leave the shifted index unaffected or adjust it up or down, depending on the calculated distance differences. In the case of doubling the number of angles, this adjustment is -1, 0 or 1. Because of the shift operation, the number of angles can only be increased in powers of 2. A block diagram of the interpolation unit is shown in Fig. 3.10, with the input muxes left out for clarity.

3.2.7 Unwrapping

The symmetry of the QAM constellation points makes it possible to limit the test phases to between 0 and $\pi/2$, which reduces hardware complexity but creates phase-wrapping problems. When the carrier phase of the signal reaches a value lower or higher than these limits, the detected phase angle will wrap around. To handle these situations, an unwrapping unit is inserted in the design. The task of this unit is to keep track of which quadrant the phase is currently in and output an unwrapped index that is composed of the interpolated index concatenated with two bits representing the quadrant.

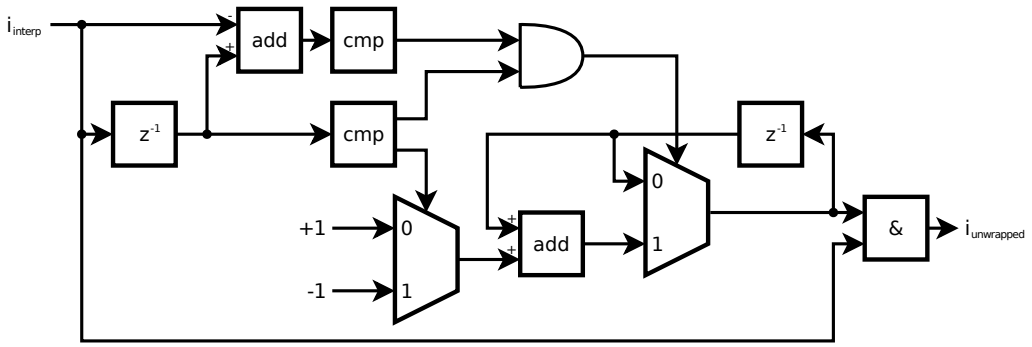


Figure 3.11: Block diagram of the unwrapping module.

A block diagram of the unwrapping DSP unit is shown in Fig. 3.11. The design calculates the difference between the current index and the previous one, creating a measure of how much the phase has changed from one sample to the next. If the change is more than $\pi/4$, or half the number of test phases, wrapping has occurred. The previous index is used to determine if the phase has changed to a higher or lower quadrant and a register is used to store the current quadrant. If the previous angle is larger than $\pi/4$ positive wrapping has occurred, while the opposite is the case for angles smaller than $\pi/4$.

Failure to detect a phase wrapping, or if a wrapping is erroneously detected, results in the estimated phase being offset by a multiple of $\pi/2$. These types of errors are called cycle slips and cause all of the following symbols to be decoded incorrectly. It is virtually impossible to recover from a cycle slip [12].

3.2.8 Compensation

Removal of the carrier phase from the input symbols is performed in the compensation unit, which also works as the slave in a master-slave configuration. Multiple samples are handled in parallel and one such parallel track is shown in Fig. 3.12. The input data is delayed to match the pipelining of the carrier estimation and rotation of the input symbols is performed as a complex multiplication. As one constant is used per sample, multipliers are a more hardware-efficient choice than, e.g., MCM. The polar form of the compensation angle is taken from a LUT, which is addressed by the unwrapped angle index, i.e. both quadrant and index.

Circular buffers are used to delay the input signal. These buffers are more power efficient than, e.g., a shift register implementation, because of the reduced switching activity. If clock gating is added as well, the switching of the flip flops in the buffer is further limited by stopping unnecessary clock activity. The registers added in front of the multipliers represent an additional pipelining stage, which minimizes the path through the multipliers. This path would otherwise be relatively long, forcing the tool used to synthesize the design to a standard-cell netlist to increase the size of the gates in the multiplier, which in turn would increase the power consumption.

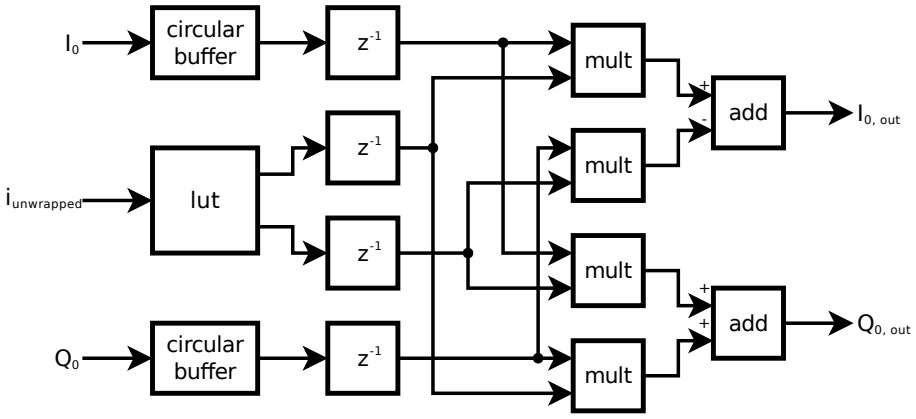


Figure 3.12: Overview of the DSP block used for carrier-phase compensation based on the phase angle calculated by the estimation block.

4

Results

This chapter presents results from simulations of the design proposed in Chapter 3. Specifically, we will present different parameter settings, such as the type of average calculation, average block size and number of internal bits, and the impact these have on the BER. The results of synthesizing the design to a standard-cell netlist are also shown, to get an estimation of the area and power needs of the design. Finally a placed and routed version of the BPS algorithm is presented and evaluated.

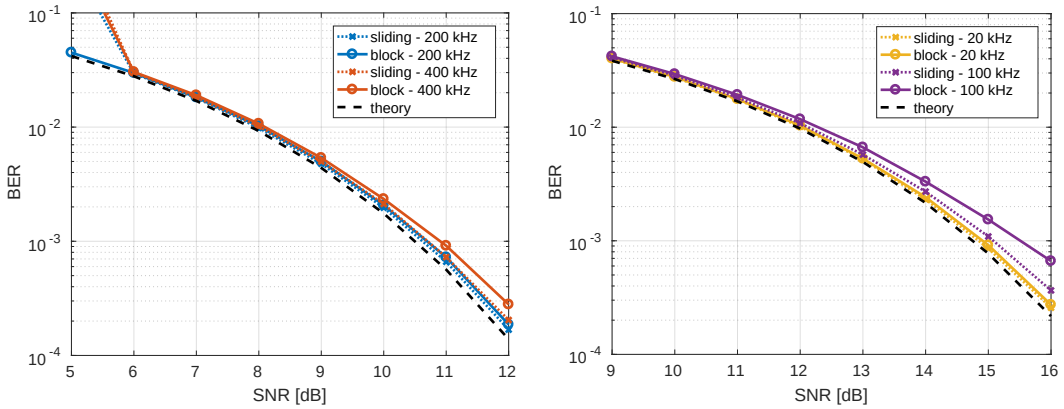
4.1 Floating-Point Simulations

Floating-point simulations were carried out on a MATLAB BPS implementation to study how the modifications to the average calculations affect the result. The MATLAB model was also used to study how the distance magnitude approximation affects the BER. A symbol rate of 20 GBd and 10^5 test symbols were used for all of these simulations.

4.1.1 Block Average

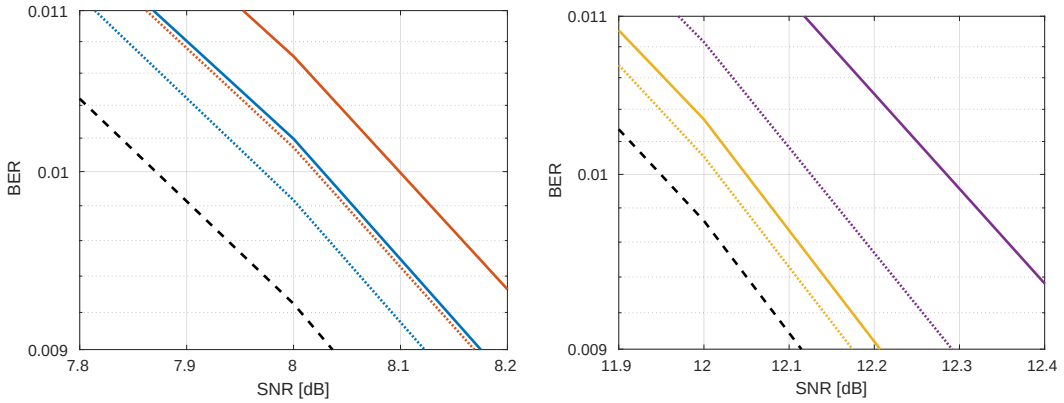
One of the simplifications made in the design process was to replace the sliding window average with a block average. To study how this change affects the BER, simulations of the MATLAB floating-point implementation were performed with both average types and varying carrier phase noise. The simulations use 8 test phases and an average block size of 64 for 16QAM. For the 64QAM version, 16 test phases and a block size of 128 were used. The number of test phases are based on the work done in [16].

Results from the simulations are shown in Fig. 4.1, with details for BER=10^{-2} shown in Fig. 4.2. The penalty, i.e. the increase in SNR needed to reach the same BER, of using a block average instead of a sliding window for 16QAM is approximately 0.05 dB at a linewidth of 20 kHz and 0.03 dB at 200 kHz for 64QAM. The penalty increases with the linewidth and is more pronounced at high SNRs, where phase noise is the dominating source of bit errors. Similar results have previously been shown for phase recovery of a 16QAM signal using QPSK partitioning [25]. For signals with a larger phase noise, the penalty induced by using a block average might be too large. The implementation cost of a parallel sliding window average is, however, very high, which can offset the potential penalty gains.



(a) 16QAM **(b)** 64QAM

Figure 4.1: Comparison of the BER impact of sliding and block average calculations.



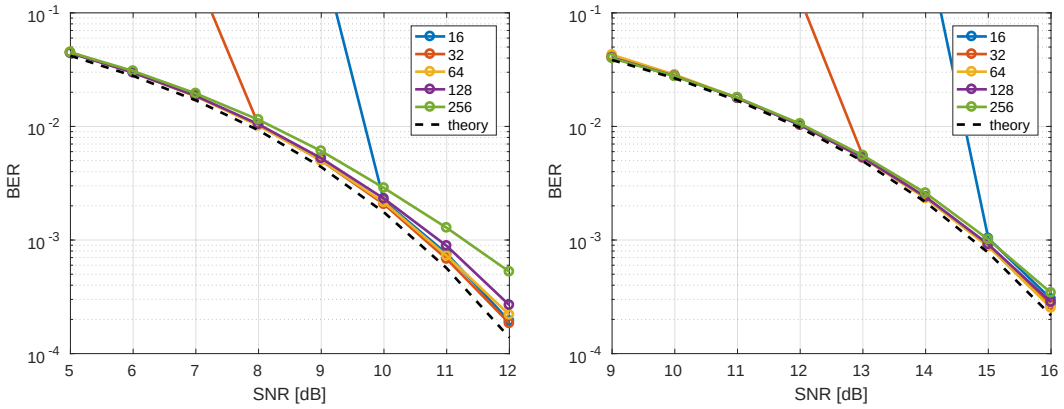
(a) 16QAM **(b)** 64QAM

Figure 4.2: Details of comparisons between sliding window and block average, centered around BER=10^{-2}. The line colors are the same as in Fig. 4.1 and the axes have been adjusted to differentiate more easily between the curves.

4.1.2 Block Size

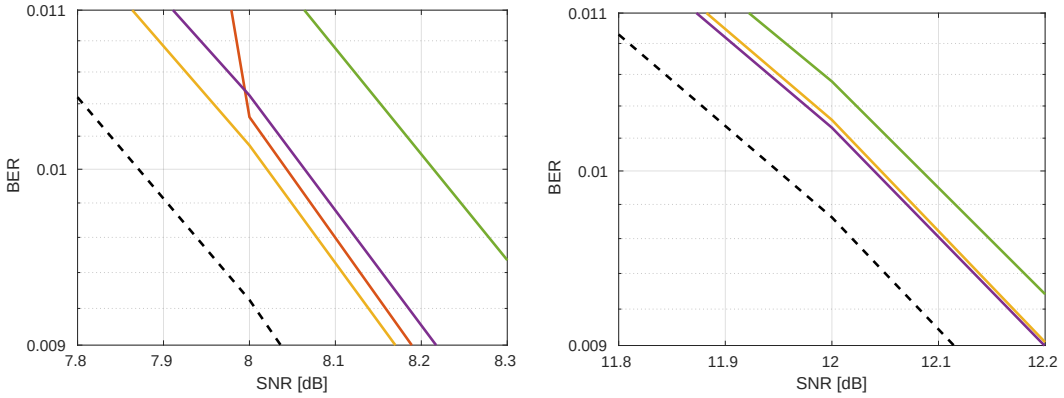
The size of the averaging block affects the system's BER, as a longer block will be more efficient in reducing the impact of white Gaussian noise. An increased block length will, however, make the system less efficient in handling fast carrier phase changes. Results from simulations using the block averaging method with varying block lengths are shown in Fig. 4.3, with details of the area around BER=10^{-2} shown in Fig. 4.4. These simulations use 8 test phases for 16QAM and 16 test phases for 64QAM. The linewidth was set to 200 kHz and 20 kHz for 16QAM and 64QAM, respectively

The results show that the two shorter block lengths, 16 and 32 samples, are too short to efficiently filter out the white noise at low SNRs, which cause cycle slips



(a) 16QAM **(b)** 64QAM

Figure 4.3: BER as a function of SNR for varying block sizes.



(a) 16QAM **(b)** 64QAM

Figure 4.4: Details of Fig. 4.3 showing the effect of varying average block sizes, centered around BER=10^{-2}. The line colors are the same as in Fig. 4.3 and the axes have been adjusted to better show the penalties involved.

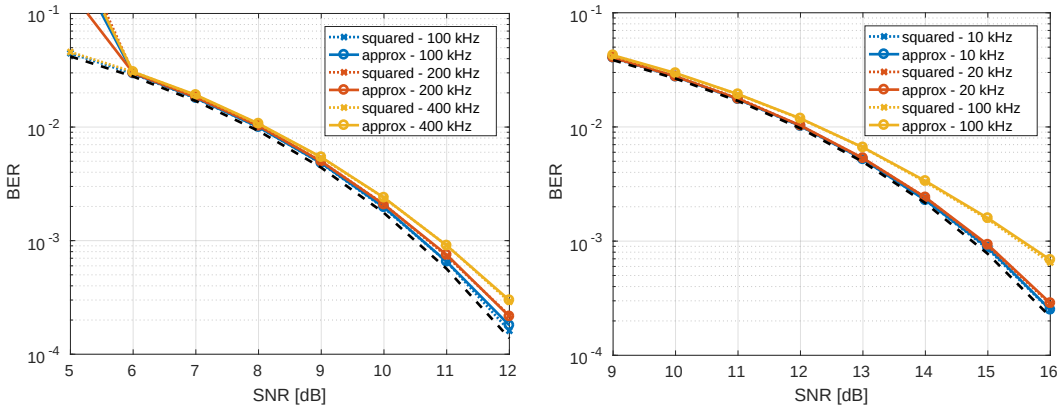
to occur for both 16QAM and 64QAM. At higher SNRs, the longer block lengths exhibit a higher penalty, an effect that is especially pronounced for 16QAM because of the greater linewidth used for these simulations. A shorter block length results in a smaller design and good block length choices, considering both white noise and phase noise, are 64 samples for 16QAM and 128 samples for 64QAM.

4.1.3 Magnitude Approximation

To study how the magnitude approximation described in Section 3.2.3 affects the error rate of the BPS algorithm, floating-point simulations were performed using both the original calculation of the squared magnitude, presented in Section 2.3, and the approximation. Three different linewidths were used for both modulation formats and the number of test phases were 8 and 16 for 16QAM and 64QAM, respectively. The simulations used a block length of 64 for 16QAM and 128 for

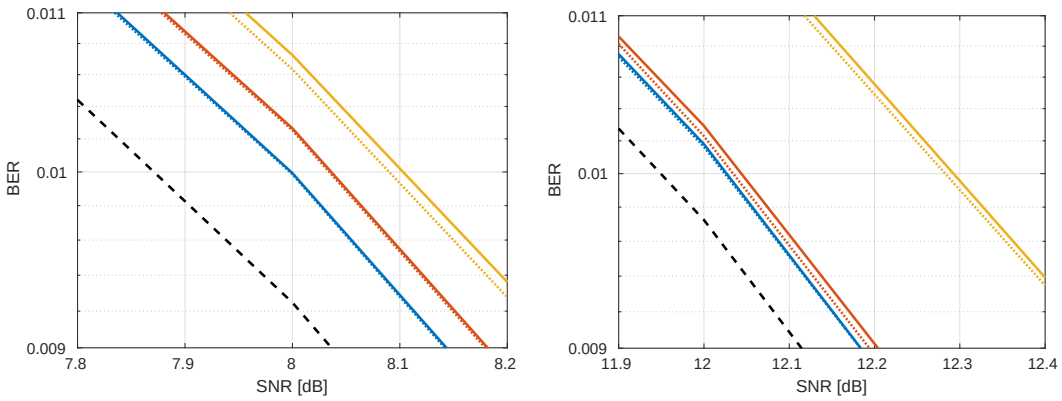
64QAM. The results are shown in Fig. 4.5, with details of the area around BER=10^{-2} shown in Fig. 4.6.

From these figures, it is clear that the approximation has a minimal effect on the resulting BER for the floating-point implementation. It is, however, possible that the magnitude approximation has a larger impact on fixed-point calculations, as one bit of data is shifted out in the division.



(a) 16QAM **(b)** 64QAM

Figure 4.5: The effect of approximate calculation of the distance magnitude on BER.



(a) 16QAM **(b)** 64QAM

Figure 4.6: Details of how the magnitude approximation affects BER in the area around BER=10^{-2}.

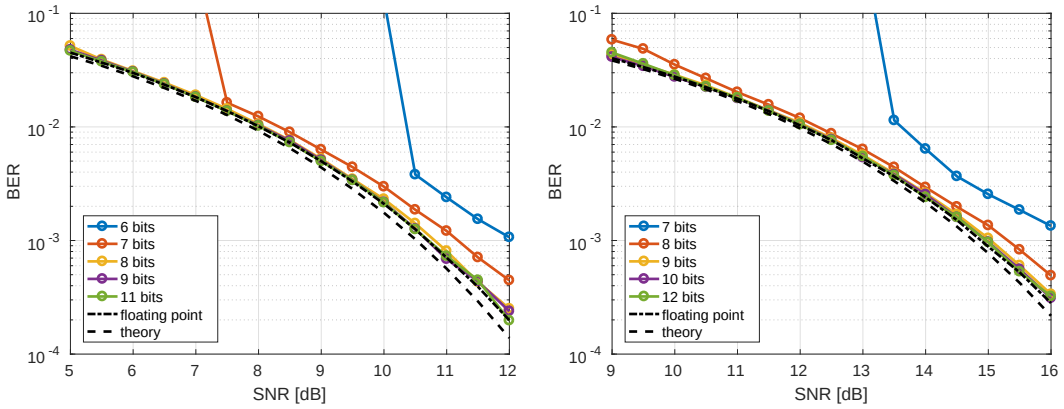
4.2 Fixed-Point Simulations

To study how the fixed-point VHDL implementation of BPS compares to the floating-point version, simulations were performed using Cadence Incisive. Test vectors were generated in MATLAB, processed by the VHDL design and read back into MATLAB for presentation. The simulations study how the number of bits used to represent

the two main data types in the design, the input and the magnitude of the distance to the closest constellation point, affects the BER. A smaller number of bits results in a smaller area footprint and less power consumption, but has a negative effect on the BER. Thus, the bits should be kept as few as possible without incurring a significant penalty compared to the floating-point implementation. All of the following simulations use 256 000 test symbols at a symbol rate of 20 GBd.

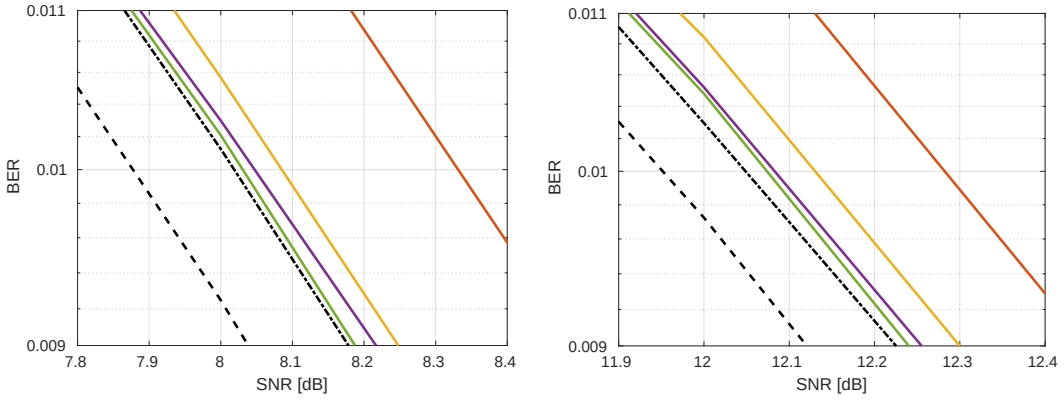
4.2.1 Number of Input Bits

The resolution of the input, i.e. the number of bits used (N), strongly affects the BER of the system and results from logic simulations with varying input word lengths are shown in Fig 4.7. The distances to the closest constellation point use an unsigned representation and $N-1$ bits are used for these signals, which eliminates the possibility of saturation of the magnitude values. As in previous simulations, the number of test phases was 8 for 16QAM and 16 for 64QAM.



(a) 16QAM **(b)** 64QAM

Figure 4.7: BER as a function of SNR for varying input word lengths.



(a) 16QAM **(b)** 64QAM

Figure 4.8: Details of input word length simulations. The axes have been adjusted to better show the penalty.

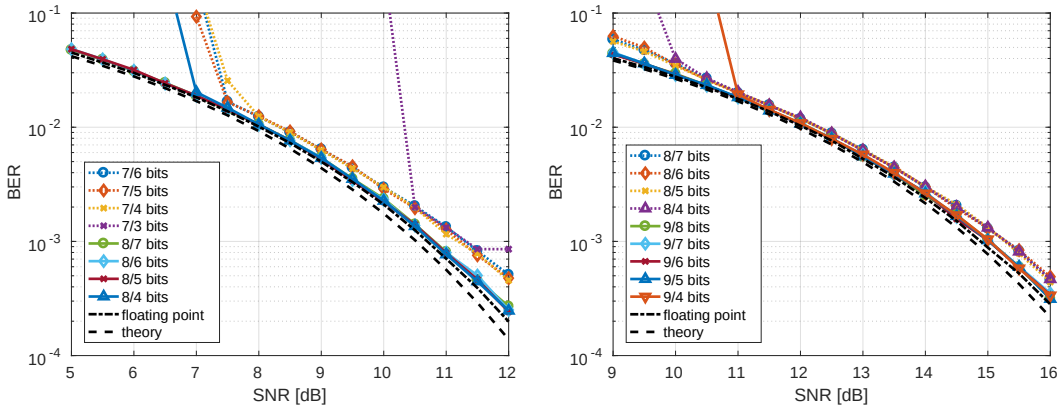
The results show that using too few input bits results in inadequate performance at higher noise levels, causing cycle slips. For 16QAM the 6-bit version stops working at an SNR lower than 11.5 dB and the same is true for the 64QAM 7-bit version at 13.5 dB. The 7-bit 16QAM curve fails under 7.5 dB, but at that point the BER is lower than the target of 10^{-2}.

Detailed views of the results around BER=10^{-2} are shown in Fig. 4.8. They show that increasing the number of bits reduces the penalty compared to the floating point versions, but that the improvement is smaller for each additional bit. Based on these simulations, 7 and 8 bits were chosen for further testing of the 16QAM implementation and 8 and 9 bits were chosen for 64QAM. The fact that the penalty for the 11-bit fixed-point implementation is very low, indicates that the effect of the magnitude approximation is minimal also for fixed-point calculations.

4.2.2 Number of Distance Bits

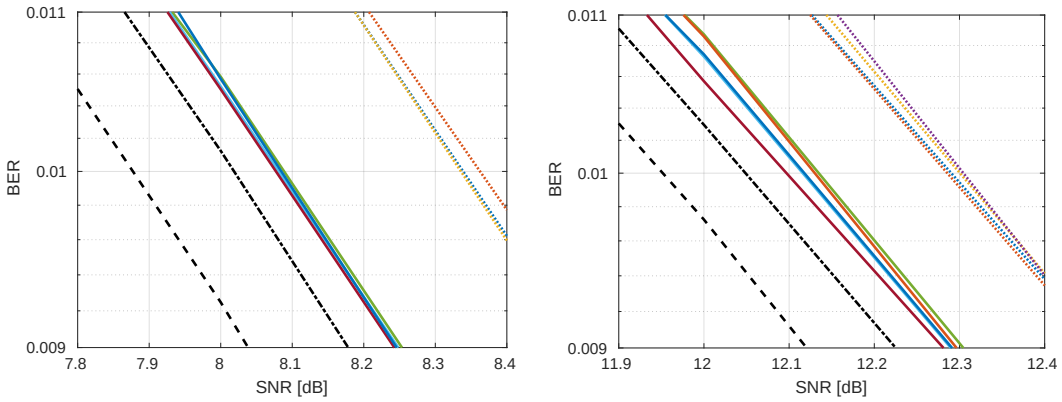
To study the effect that the word length of the calculated distance to the closest constellation point, i.e. the output from the distance to closest symbol unit in Fig. 3.2, has on the system's BER, logic simulations for both 16QAM and 64QAM were performed. The same settings were used as for the simulation above and the results are shown in Fig 4.9. Details of the area surrounding BER=10^{-2} in these plots are shown in Fig. 4.10. The curves are labeled N/M, where N is the number of input bits and M is the number of magnitude bits.

The number of bits used to represent the magnitude does not have a significant impact on the BER, as long as there are enough bits to avoid cycle slips. For 16QAM, these cycle slips occur for the 7/3 and the 8/4 version. The same type of failures can be seen for 64QAM using the 8/4 and 9/4 configurations. The failures of the other 7 bit curves for 16QAM are most likely due to the limited resolution of the input bits, as seen in the previous section. Based on these simulations, 5 bits were selected to represent the distance in further simulations as this was the minimum number of bits still producing correct results.



(a) 16QAM **(b)** 64QAM

Figure 4.9: BER as a function of SNR for varying word lengths of the distance to the closest constellation point.

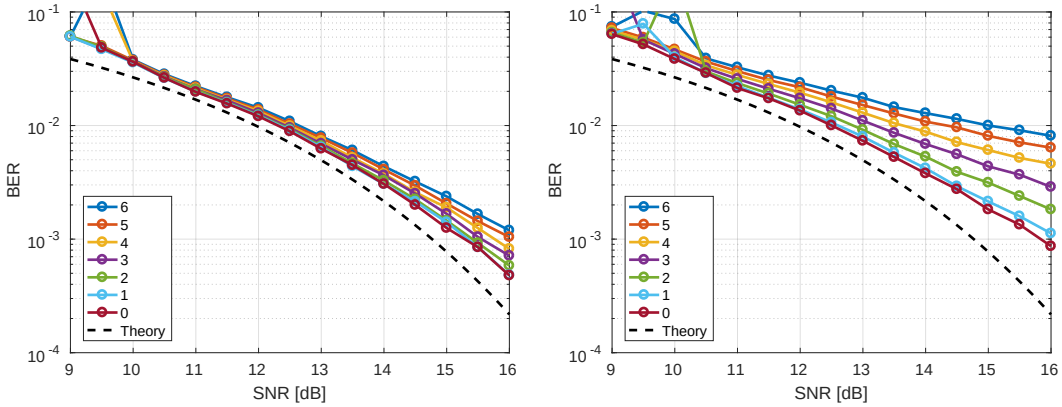


(a) 16QAM **(b)** 64QAM

Figure 4.10: Detail of the area surrounding BER=10^{-2} in Fig 4.9. The axes have been adjusted to better show the penalty.

4.2.3 Delay of the Estimated Carrier Phase

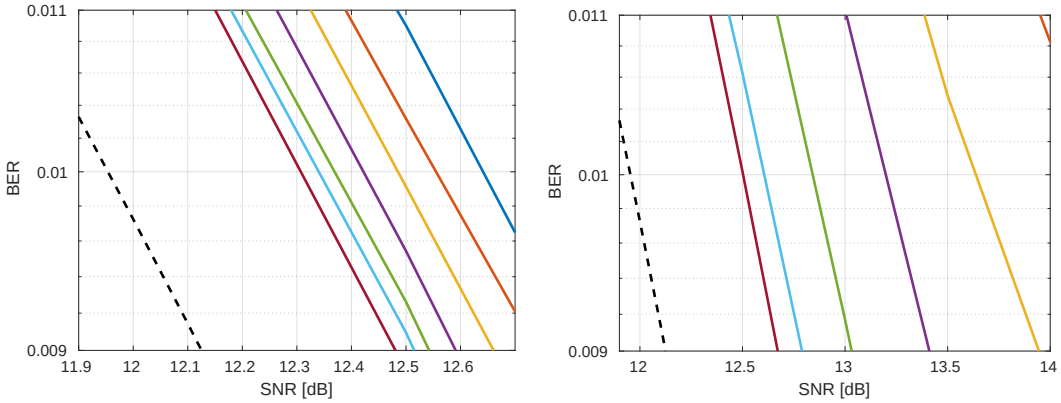
When placing a compensation unit on the die, it might need to be situated some distance away from the estimation unit, especially if multiple compensation units are driven by the same estimation unit. If this separation is too great, it can be necessary to insert a pipeline register in the signal paths carrying the estimated phase, which will incur a latency. It is possible to add another delay stage in the compensation unit in order to keep the synchronization, but this adds to the latency of the compensation and increases power consumption and area utilization, especially since this part of the design is highly parallel. To study how a delayed estimated phase affects the results, simulations of a 64QAM implementation were carried out with varying delays. The results are shown in Fig. 4.11 with details in Fig. 4.12. Similar results have been shown for 16QAM.



(a) 20 kHz **(b)** 100 kHz

Figure 4.11: Results of simulations for varying delays of the estimated phase angle.

The penalty of a larger delay is more prominent at higher SNRs, as phase noise is the main cause of errors at lower noise levels. At higher linewidths the effect is even larger. Inserting a register in the estimated signal path, i.e. creating a delay of one clock cycle, does not result in a major penalty at lower linewidths, indicating that this method might be feasible if the cost of adding an extra delay stage to the compensation unit is too high.



(a) 20 kHz **(b)** 100 kHz

Figure 4.12: Details of simulations for varying delays of the estimated phase angle. The axes have been adjusted to better illustrate the penalties involved.

4.3 Synthesis Results

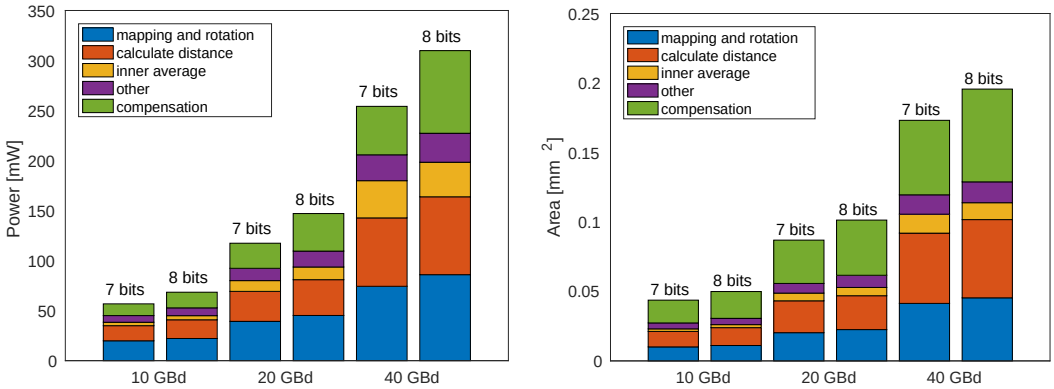
Synthesis of the design followed by a simulation-based power analysis provides good estimates of cell area and power consumption. Synthesis was performed using Cadence Genus and a 28 nm STMicroelectronics, regular-V_t, low-power cell library. A slow-slow (SS) process corner, an operation temperature of 125 °C and a supply voltage of 0.9 V were used. The syntesized netlist was simulated in Cadence Incisive and the resulting switching activity interchange file (SAIF) was backannotated into Genus to supply switching statistics for the power analysis. Both insertion of clock-gating logic and retiming of the design were performed during synthesis.

4.3.1 16QAM

Two 16QAM versions were synthesized with parameters selected based on the previously described simulations. The versions differ in the number of input bits, 7 and 8, while the other parameters were held constant. The word length of the distance to the closest constellation point was set to 5 bits, the block size to 64 and the number of test phases to 8. The clock rate used was 1600 ps and the baud rates tested were 10, 20 and 40 GBd, which corresponds to a parallelization factor of 16, 32 and 64, respectively.

Table 4.1: Synthesis results for 16QAM using 8 test phases, an averaging block size of 64 and 5 bits for the distance to the closest constellation point.

Input Bits	10 GBd		20 GBd		40 GBd	
	Power [mW]	Area [µm^2]	Power [mW]	Area [µm^2]	Power [mW]	Area [µm^2]
7	56.79	43727	117.40	86919	254.19	173128
8	68.43	49916	147.05	101339	309.87	195611



(a) Power **(b)** Area

Figure 4.13: Resource distribution for 16QAM.

Statistics of the cell area and power consumption of the design are presented in Table 4.1, with the distribution among the major DSP units shown in Fig. 4.13. For detailed tables of the power and area distributions, see Appendix A. The scaling of both power and area with increased symbol rate is close to linear for both input widths. An exception is the power for both 40-GBd versions, which consume 4.4 times more power than the 10-GBd versions. This increase is most likely due to a much larger adder in the inner average unit, which increases the power consumption and forces the synthesis tool to take other retiming decisions than for the other versions.

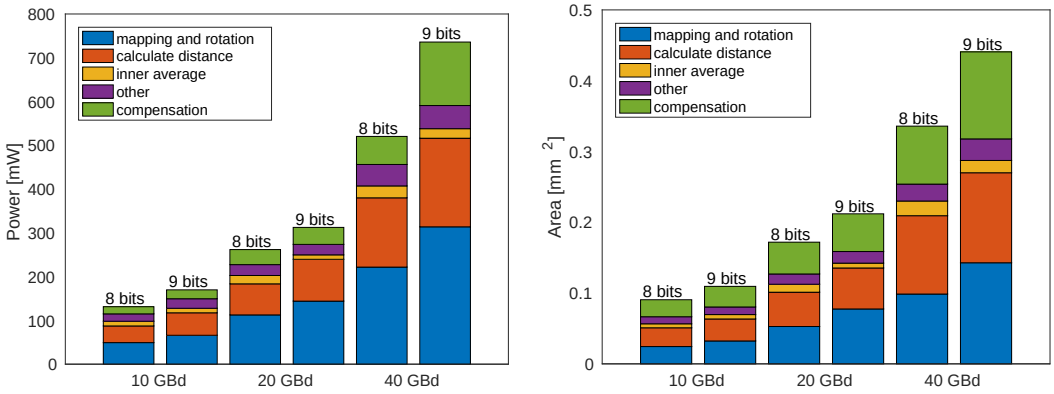
Adding an extra bit of input resolution to the design we get a 20 % increase in power consumption and a 14 % to 23 % area increase, where the larger value is for the 40-GBd design. The compensation unit consumes 20 % of the total power and takes up 35 % of the total cell area for the 7-bit versions, with numbers being slightly higher for the 8-bit design, most probably due to the more complex multipliers needed.

4.3.2 64QAM

For 64QAM the number of test phases was set to 16, the block size to 128, the word length of the distance to closest constellation point to 5 bits and the two input bit widths tested were 8 and 9. The same clock rate and parallelization factors apply as for the 16QAM versions. Synthesis results are presented in Table 4.2 and the

Table 4.2: Synthesis results for 64QAM using 16 test phases, an averaging block size of 128 and 5 bits for the distance to the closest constellation point.

Input Bits	10 GBd Power [mW]	10 GBd Area [µm²]	20 GBd Power [mW]	20 GBd Area [µm²]	40 GBd Power [mW]	40 GBd Area [µm²]
8	131.21	90598	261.77	171894	520.31	335816
9	169.79	106476	312.46	211947	735.90	440683



(a) Power **(b)** Area

Figure 4.14: Resource distribution for 64QAM.

resource distribution among the DSP units are shown in Fig. 4.14.

The scaling with increased baud rate behaves similarly to the 16QAM case and is close to linear. With the increased number of test phases and constellations points used for 64QAM, the inner average is now proportionally a much smaller part, which makes the impact of the larger adder insignificant.

The addition of an additional input bit induces a power increase of between 20 % and 40 %, with the larger value for the 40-GBd version. The extra added power for the faster version is mostly due to a much larger compensation unit, presumably caused by the scaling of the multipliers when adding an extra bit. The power consumption of the multipliers has a larger impact at the higher baud rates due to the higher parallelization factor.

4.3.3 Energy/Bit

A common way to evaluate a DSP design is to calculate the energy consumption per processed bit; such values for the synthesized implementations can be found in Table 4.3. The table shows that the cost of increasing the input bit width is between 19 and 40 % and an increase in the symbol rate does not necessarily incur a cost in terms of energy/bit. The constant energy/bit values with increased baud rate were more pronounced before retiming of the designs, as the less parallelized versions

Table 4.3: Energy consumption per bit for synthesized designs.

	input bits	Energy/bit [pJ/bit] 10 GBd	20 GBd	40 GBd
16QAM	7	1.42	1.47	1.59
	8	1.71	1.84	1.94
64QAM	8	2.19	2.18	2.17
	9	2.83	2.60	3.07

responded better to retiming than the larger ones.

To put the values presented in Table. 4.3 in perspective, they can be compared to e.g. the carrier recovery implementation presented by Crivelli et al. [6]. Their system uses QPSK encoding and the carrier recovery consumes 0.717 W at 12.5 GBd resulting in an energy/bit value of 28 pJ/bit. Their system was however realized using a 40 nm process. Given that the power reduction can be estimated to approximately 30 % for each smaller process node [26], the power consumption would be roughly 20 pJ/bit at 28 nm. Another reference value can be taken from Pillai et al. [27], who suggest a power consumption of 16 pJ/bit for QPSK carrier recovery in their model of a long-haul coherent fiber-optical system. They also budget approximately 5.5 times more power for PMD compensation than for carrier recovery, and a dynamic equalizer implemented using the 28 nm process node has been shown to consume 18 pJ/bit [28].

4.4 Place and Route

The process of place and route was performed using the Cadence Innovus system. After place and route, more accurate values for the power consumption and area can be obtained, since the properties of all wiring in the system is now known. A 64QAM BPS implementation using one estimation unit and two compensation units was synthesized in Cadence Genus. The system settings were the same as for the 64QAM 20-GBd implementation described above, using 8 bits for the input width. The hierarchies of all submodules to the estimation and compensation units were flattened during synthesis, to take full advantage of the tool's optimizations and to facilitate retiming.

Table 4.4: Power and area of a 64QAM design after place and route

Component	Post-Synthesis Power [mW]	Area [µm^2]	Post-Place & Route Power [mW]	Area [µm^2]
estimation	198.71	129769	184.42	136781
compensation0	34.32	44672	35.27	46881
compensation1	34.99	45068	34.46	45652
total	268.03	219521	256.29	229649

The final distribution of the components on the die is shown in Fig. 4.15 and the power and cell area, both after synthesis and place and route, are shown in Table 4.4. It is apparent that the flattening of hierarchies allowed the synthesis tool to optimize the design more efficiently than before, since the extra compensation unit is almost for free compared to the values in Table. 4.2. The design with one estimation and two compensation units reduces the energy/bit value to 1.7 pJ/bit, with further reductions possible as more slave compensation units are added. However, the slack of the path between the master estimation unit and the slaves limits the number of units that can be added.

The post-synthesis power consumption is also slightly higher than that of post-place and route. This indicates that the design was easier to route than expected, and that it might be possible to use a smaller die size. The current die core area is 0.366 mm^2, resulting in a utilization of 63 %

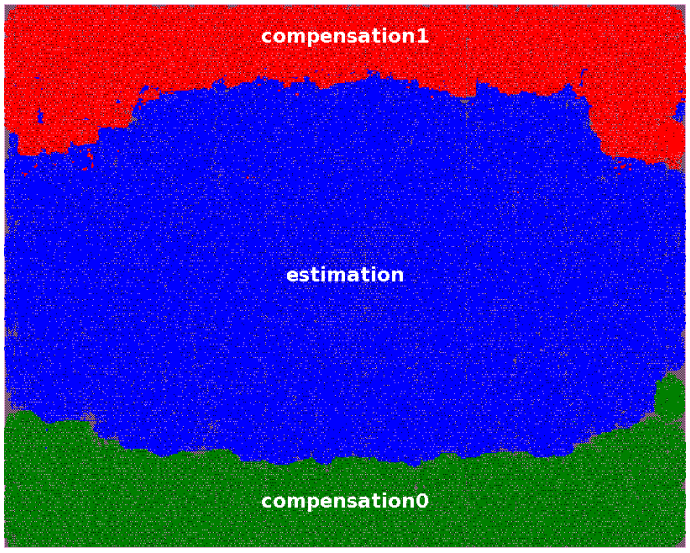


Figure 4.15: Component distribution for a 64QAM implementation after place and route. The design uses 8 input bits, 5 bits for the distance to the closest constellation point, 16 test phases, an averaging block size of 128 and a symbol rate 20 GBd.

5

Conclusion

In this thesis we have proposed, described and evaluated a VHDL implementation of BPS as a phase-recovery method in coherent fiber-optical systems. The design has been optimized mainly with power consumption in mind, but with the aim of not adding a severe BER penalty compared to a floating-point implementation. An evaluation of the impact of using a fixed-point representation has been performed and the effects of the optimizations and approximations used in the implementation have been studied. Data of power consumption and area have been presented, both for synthesized designs and for a placed and routed version. We have shown that an ASIC implementation of BPS is possible and that the cost, in terms of power and area, is not prohibitively high.

The proposed implementation of BPS has been shown to work for 16 and 64QAM, with a penalty of less than 0.3 dB compared to a floating-point implementation, for baud rates up to 40 GBd. It is also possible to scale the design upwards to higher baud rates by more extensive parallelization, up to a level where the parallelization factor equals the averaging block length. Some of the techniques used to reduce power consumption, such as MCM, mapping of the symbols to the first quadrant and clock gating, do not affect the quality of the output. Other parameters, such as bit widths, can be selected to find a balance between error rate and power consumption. The approximation used to calculate the magnitude does not only reduce the number of multipliers needed, but also limits the number of bits in the following stages, without significantly increasing the BER.

The use of block averaging instead of a sliding window works at relatively benign linewidths. A higher phase noise causes the penalty to increase and this is where we see most of the potential for improvement. With a sliding window average, the system would have a higher phase-noise tolerance, which would relax the requirements on the lasers. The implementation of such a solution must, however, be carefully considered to reduce the impact on latency and power consumption.

We have shown that our BPS implementation can be a feasible alternative to pilot-tone-based methods for carrier recovery in a fiber-optical systems, if the linewidth of the lasers are low enough. By eliminating the pilot tone, more bandwidth will be available for data transmission and higher data throughput might be possible. The laser linewidth is directly related to the quality of the lasers and the trade-off between hardware cost and potential increase in data throughput needs to be considered when designing a complete system. The reduced risk of cycle-slips in a pilot-tone-based approach and the fact that other parts of the DSP chain can

use the pilot tone, e.g. the frequency-recovery unit, are also factors to take into consideration.

The placed and routed design shows that a master-slave implementation of the BPS algorithm is feasible, as long as the timing of the signals transmitted between the modules is met. A delay of the estimate phase angle of one sample might be used without severely affecting the results, at least for systems with low phase noise.

Suggestions for future work, apart from possibly implementing a sliding window averaging, can include synthesizing designs for higher modulations formats, such as 256QAM, to further study how the design scales. Synthesis of a system with more than two compensation units is also a promising path, as it will reduce the energy/bit significantly. The maximum number of slaves that can be used for each estimation unit will also need further investigation, as the path length is a limiting factor. Insertion of the carrier recovery inside the feedback loop of a decision-directed equalizer can also be considered, but a reduction of the latency of the design might be needed to make this possible.

Bibliography

[1] D. C. Kilper, G. Atkinson, S. K. Korotky, S. Goyal, P. Vetter, D. Suvakovic, and O. Blume, "Power trends in communication networks," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 17, no. 2, pp. 275–284, March 2011.

[2] L. Lundberg, M. Mazur, A. Lorences-Riesgo, M. Karlsson, and P. A. Andrekson, "Joint carrier recovery for DSP complexity reduction in frequency comb-based superchannel tranceivers," unpublished.

[3] G. P. Agrawal, *Fiber-optic communication systems*, 4th ed. New York: Wiley, 2010.

[4] X. Liu, S. Chandrasekhar, and P. J. Winzer, "Digital signal processing techniques enabling multi-Tb/s superchannel transmission: An overview of recent advances in DSP-enabled superchannels," *IEEE Signal Processing Magazine*, vol. 31, no. 2, pp. 16–24, March 2014.

[5] E. Ip, A. P. T. Lau, D. J. F. Barros, and J. M. Kahn, "Coherent detection in optical fiber systems," *Opt. Express*, vol. 16, no. 2, pp. 753–791, Jan 2008.

[6] D. E. Crivelli, M. R. Hueda, H. S. Carrer, M. del Barco, R. R. López, P. Gianni, J. Finochietto, N. Swenson, P. Voois, and O. E. Agazzi, "Architecture of a single-chip 50 Gb/s DP-QPSK/BPSK transceiver with electronic dispersion compensation for coherent optical channels," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 4, pp. 1012–1025, 2014.

[7] S. J. Savory, "Digital coherent optical receivers: Algorithms and subsystems," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 16, no. 5, pp. 1164–1179, 2010.

[8] T. Tanimura, S. Oda, T. Tanaka, T. Hoshida, Z. Tao, and J. C. Rasmussen, "A simple digital skew compensator for coherent receiver," in *2009 35th European Conference on Optical Communication (ECOC)*, Sept 2009, pp. 1–2.

[9] B. E. A. Saleh and M. C. Teich, *Fundamentals of photonics*, 2nd ed. Chichester; Hoboken, N.J: Wiley-Interscience, 2013.

[10] S. J. Savory, "Digital filters for coherent optical receivers," *Optics Express*, vol. 16, no. 2, pp. 804–817, 2008.

[11] K. Kikuchi, "Clock recovering characteristics of adaptive finite-impulse-response filters in digital coherent optical receivers," *Opt. Express*, vol. 19, no. 6, pp. 5611–5619, Mar 2011.

[12] L. Lundberg, "Aspects of power consumption in coherent fiber-optical communication systems," Lic. thesis, Chalmers University of Technology, Gothenburg, Sweden, 2017.

[13] T. Pfau, S. Hoffmann, and N. Reinhold, "Hardware-efficient coherent digital receiver concept with feedforward carrier recovery for M-QAM constellations," *Journal of Lightwave Technology*, vol. 27, no. 8, pp. 989–999, 2009.

[14] M. Morsy-Osman, Q. Zhuge, L. R. Chen, and D. V. Plant, "Feedforward carrier recovery via pilot-aided transmission for single-carrier systems with arbitrary M-QAM constellations," *Opt. Express*, vol. 19, no. 24, pp. 24 331–24 343, Nov 2011.

[15] A. Viterbi and A. Viterbi, "Nonlinear estimation of PSK-modulated carrier phase with application to burst digital transmission," *IEEE Transactions on Information Theory*, vol. 29, no. 4, pp. 543–551, 1983.

[16] H. Sun, K. T. Wu, S. Thomson, and Y. Wu, "Novel 16QAM carrier recovery based on blind phase search," in *2014 The European Conference on Optical Communication (ECOC)*, Sept 2014, pp. 1–3.

[17] C. Liu, J. Pan, T. Detwiler, A. Stark, Y.-T. Hsueh, G.-K. Chang, and S. E. Ralph, "Joint digital signal processing for superchannel coherent optical communication systems," *Opt. Express*, vol. 21, no. 7, pp. 8342–8356, Apr 2013.

[18] M. D. Feuer, L. E. Nelson, X. Zhou, S. L. Woodward, R. Isaac, B. Zhu, T. F. Taunay, M. Fishteyn, J. M. Fini, and M. F. Yan, "Joint digital signal processing receivers for spatial superchannels," *IEEE Photonics Technology Letters*, vol. 24, no. 21, pp. 1957–1960, 2012.

[19] R. G. H. van Uden, C. M. Okonkwo, V. A. J. M. Sleiffer, M. Kuschnerov, H. de Waardt, and A. M. J. Koonen, "Single DPLL joint carrier phase compensation for few-mode fiber transmission," *IEEE Photonics Technology Letters*, vol. 25, no. 14, pp. 1381–1384, July 2013.

[20] K. K. Parhi, *VLSI Digital Signal Processing Systems*. Chichester; Hoboken, N.J: Whiley-Interscience, 1999.

[21] Y. Voronenko and M. Püschel, "Multiplierless multiple constant multiplication," *ACM Trans. Algorithms*, vol. 3, no. 2, May 2007.

[22] R. G. Lyons, *Understanding Digital Signal Processing*, 2nd ed. Upper Saddle River, N.J: Prentice Hall PIR, 2004.

[23] *ChipWare IP Components in Encounter RTL Compiler (14.1)*, Cadence Design Systems, 2014.

[24] B. Yuce, H. F. Ugurdag, S. Gören, and G. Dündar, "Fast and efficient circuit topologies for finding the maximum of n k-bit numbers," *IEEE Transactions on Computers*, vol. 63, no. 8, pp. 1868–1881, Aug 2014.

[25] I. Fatadin, D. Ives, and S. J. Savory, "Carrier phase recovery for 16-QAM using QPSK partitioning and sliding window averaging," *IEEE Photonics Technology Letters*, vol. 26, no. 9, pp. 854–857, May 2014.

[26] M. Kuschnerov, T. Bex, and P. Kainzmaier, "Energy efficient digital signal processing," in *OFC 2014*, March 2014, pp. 1–3.

[27] B. S. G. Pillai, B. Sedighi, K. Guan, N. P. Anthapadmanabhan, W. Shieh, K. J. Hinton, and R. S. Tucker, "End-to-end energy modeling and analysis of long-haul coherent transmission systems," *Journal of Lightwave Technology*, vol. 32, no. 18, pp. 3093–3111, Sept 2014.

[28] C. Fougstedt, P. Johannisson, L. Svensson, and P. Larsson-Edefors, "Dynamic equalizer power dissipation optimization," in *2016 Optical Fiber Communications Conference and Exhibition (OFC)*, March 2016, pp. 1–3.

A

Appendix 1

Table A.1: Synthesis results for 16QAM with 7 input bits. The total values include the pipeline registers present in the design.

	10 GBd		20 GBd		40 GBd	
	Power [mW]	Area [µm²]	Power [mW]	Area [µm²]	Power [mW]	Area [µm²]
mapping and rotation	19.88	10081	39.30	20313	74.33	41374
calculate distance	15.11	11122	29.95	22932	68.37	50559
inner average	3.40	1860	10.77	5541	37.20	13641
outer average	0.62	759	0.29	314		
minimum average	0.03	119	0.03	118	0.16	138
interpolation	0.06	212	0.11	275	0.16	227
unwrapping	0.02	43	0.05	57	0.05	36
compensation	11.71	16405	25.09	31174	48.43	53607
total	56.79	43727	117.40	86919	254.19	173128

Table A.2: Synthesis results for 16QAM with 8 input bits. The total values include the pipeline registers present in the design.

	10 GBd		20 GBd		40 GBd	
	Power [mW]	Area [µm²]	Power [mW]	Area [µm²]	Power [mW]	Area [µm²]
mapping and rotation	22.17	11077	45.18	22533	85.96	45390
calculate distance	18.70	12856	35.78	24380	77.84	56297
inner average	4.11	2170	12.66	5944	34.53	12154
outer average	0.80	760	0.31	314		
minimum average	0.05	119	0.07	115	0.38	136
interpolation	0.11	210	0.18	282	0.46	339
unwrapping	0.02	43	0.04	42	0.07	38
compensation	15.64	19291	37.54	39721	82.58	66767
total	68.43	49916	147.05	101339	309.87	195611

Table A.3: Synthesis results for 64QAM with 8 input bits. The total values include the pipeline registers present in the design.

	10 GBd		20 GBd		40 GBd	
	Power [mW]	Area [µm^2]	Power [mW]	Area [µm^2]	Power [mW]	Area [µm^2]
mapping and rotation	49.25	24453	112.54	52765	221.75	98625
calculate distance	38.07	26519	70.98	48447	158.27	110682
inner average	10.70	5638	19.05	11252	27.21	20713
outer average	4.95	4001	1.71	1496	0.79	677
minimum average	0.05	323	0.09	265	0.12	301
interpolation	0.07	340	0.10	460	0.16	489
unwrapping	0.02	43	0.04	62	0.03	51
compensation	16.32	24013	34.34	44908	64.09	81898
total	131.21	90598	261.77	171894	520.31	335816

Table A.4: Synthesis results for 64QAM with 9 input bits. The total values include the pipeline registers present in the design.

	10 GBd		20 GBd		40 GBd	
	Power [mW]	Area [µm^2]	Power [mW]	Area [µm^2]	Power [mW]	Area [µm^2]
mapping and rotation	66.00	32368	144.01	77554	313.77	142823
calculate distance	51.37	31047	95.54	57898	202.40	127154
inner average	10.27	6212	10.20	6825	21.80	17368
outer average	6.65	4100	1.57	1553	0.85	656
minimum average	0.85	574	0.17	263	0.11	291
interpolation	0.20	476	0.12	487	0.17	191
unwrapping	0.02	41	0.04	63	0.03	53
compensation	20.27	29105	38.85	53259	145.04	122899
total	169.79	109476	312.46	211947	735.90	440683