VLSI Implementations of Carrier Phase Recovery Algorithms for M-QAM Fiber-Optic Systems

(article starts on next page)

VLSI Implementations of Carrier Phase Recovery Algorithms for M-QAM Fiber-Optic Systems

Erik Börjeson, *Student Member, IEEE,* Christoffer Fougstedt, and Per Larsson-Edefors, *Senior Member, IEEE*

Abstract—**We present circuit implementations of blind phase search (BPS) carrier phase recovery (CPR) for M-QAM coherent optical receivers and highlight some BPS algorithm modifications necessary to obtain efficient VLSI circuits. In addition, we show how three key design parameters (input word length, number of test phases, and type and size of averaging window) affect the resulting implementation. To study design tradeoffs, we develop BPS CPR circuit netlists for a 32-GBaud system, using a 22-nm CMOS process technology: Our implementations reach energy efficiencies of around 1 pJ/bit for 16QAM up to 3 pJ/bit for 256QAM, at an SNR penalty of approximately 0.25 dB at a BER of 10^{-2}. Furthermore, we present a circuit implementation of pilot-symbol-aided CPR, reaching 0.38 pJ/bit and 0.34 pJ/bit for 16QAM and 256QAM, respectively, at a slightly higher SNR penalty. The two CPR methods are also evaluated in terms of silicon area and scaling to higher-order modulation formats.**

I. Introduction

The ever-increasing need for higher transmission rates in the world's global communication networks puts increasingly higher demands on the fiber-optic links transporting the data. For long-haul transmissions, coherent systems have been the norm for a long time due to high receiver sensitivity and data encoding that involves amplitude, phase and two polarizations [1]. One way to increase the data transmission rate of a fiber-optic system is to increase the spectral efficiency, and this can be done by using higher-order modulation formats, such as 64QAM and 256QAM [2]. However, a challenge of these more complex modulation formats is their higher sensitivity to transmission impairments, such as additive white Gaussian noise (AWGN) and phase noise.

In coherent systems, digital signal processing (DSP) is used to compensate for many of the distortions introduced to the transmitted symbols when these are propagated through the fiber. Using advanced very-large scale integration (VLSI) techniques, the DSP is typically realized as an application-specific integrated circuit (ASIC). Since the DSP ASIC power dissipation can be a significant part of overall receiver power [3], it is essential to keep the ASIC power dissipation under control; in particular this is a challenge when moving to higher-order formats. In addition, as coherent systems are making inroads

in shorter, more cost-sensitive fiber systems, such as data-center interconnects, the constraints on silicon area and power dissipation are becoming even stricter [4].

A DSP system for fiber-optic communications is usually divided into smaller units, where each unit is responsible for compensating one or more impairments. The phase noise is handled in the carrier phase recovery (CPR) unit, where the current phase is estimated and removed from the input symbols. There are a range of different CPR algorithms to choose from, from non-data-aided (or blind) [5]–[8] to pilot-aided [9] methods. The first methods rely on the transmitted symbols to estimate the phase, while an example of the latter uses known pilot symbols, which are time-division multiplexed into the data-symbol stream to enable the estimation. For simpler modulation formats, such as quadrature phase-shift keying (QPSK), there are low-complexity blind CPR algorithms available, e.g. [5], whose VLSI circuit implementations are relatively straightforward. However, many of these algorithms are unusable for higher-order QAM transmissions, due to their multi-level amplitudes. CPR methods suggested for use with higher-order formats include QPSK-partitioning [6], maximum likelihood phase estimation [7], and multi-stage approaches combining different methods [10].

A popular CPR method for QAM transmission is the blind phase search (BPS) originally proposed for fiber-optic communication by Pfau et al. [8]. BPS uses a number of test phases to rotate the received symbols and detects which phase results in the best match to a valid constellation point. Different aspects and improvements to this algorithm have been suggested over the last years, e.g. [11], however, no VLSI circuit implementations have been presented. An investigation of how BPS can be realized in a DSP ASIC is necessary to be able to determine its implementation feasibility, both in terms of algorithm performance in a resolution-limited digital world, and in terms of silicon area and power dissipation.

In this work we present a VLSI implementation of a modified BPS algorithm. While this work is based on our recent OFC contribution [12], we here expand and elaborate on design strategies and trends for silicon area usage and power dissipation. In Section II we describe how a hardware-efficient version of the BPS algorithm can be developed, while in Section III we show how these modifications affect CPR performance. Details of the circuit design of the largest BPS circuit components are described in Section IV followed by a description of our ASIC design approach in Section V. Section VI discusses power dissipation and its dependence on parameter settings, for BPS- and pilot-based CPR methods. Finally, a conclusion is given.

E. Börjeson and P. Larsson-Edefors are with the Department of Computer Science and Engineering, Chalmers University of Technology, SE-41296 Gothenburg, Sweden (e-mail: erikbor@chalmers.se, perla@chalmers.se).

C. Fougstedt is with Ericsson Research, Gothenburg, Sweden (e-mail: christoffer.fougstedt@ericsson.com).

II. ENERGY-EFFICIENT ASIC DESIGN OF CARRIER PHASE RECOVERY ALGORITHMS

In developing VLSI circuits of DSP algorithms, algorithm modifications are most likely necessary if energy efficiency is a key design goal. For example, seemingly simple floating-point MATLAB operations of an algorithm can result in very complex VLSI implementations when realized in fixed-point digital circuits, necessitating simplifications or approximations during ASIC design. What makes ASIC design even more complex is that there are cases when the addition of a computational stage at one point in the algorithm has a large effect on the fixed-point properties of completely different parts of the corresponding VLSI circuit. Understanding these tradeoffs is essential to designing energy-efficient VLSI circuits.

A pseudo-code description of the BPS algorithm used as a base for our VLSI circuit is shown in Algorithm 1, and the following sections will describe the modifications performed on this algorithm to enable our circuit implementation, shown as a block diagram in Fig. 1. From Algorithm 1 it is clear that the circuit complexity is largely dependent on two main factors: the *number of test phases*, B, and the *size of the averaging window*, L. However, the phase-angle resolution decided by B has a large impact on the quality of the phase estimation, up to a point when increasing the number of phases further does not improve the result [8]. In addition, the choice of L affects the CPR performance, which we will discuss in Section III. A third major contributor to circuit complexity, not typically captured in algorithmic pseudo-code, is the *parallelism* necessary to reach the data-throughput demands of fiber-optic systems. Our VLSI implementation is parallelized in P lanes, which typically increases the power dissipation P times and makes certain calculations more complex, such as the summation in Algorithm 1 since this uses consecutive symbols as inputs. Finally, the choice of fixed-point resolution,
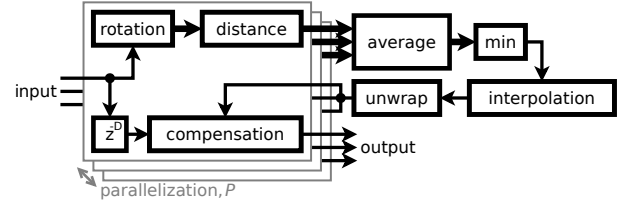

Fig. 1. Block diagram of the BPS circuit implementation, where the thick arrows represent buses carrying data from multiple rotations.
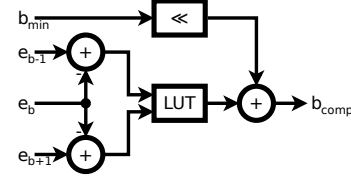

Fig. 2. Block diagram of the **interpolation** component.

or word length, at different stages in the VLSI circuit impacts both CPR performance and power dissipation.

To reduce the number of test phases, Sun et al. [11] proposed a method of parabolic interpolation, which uses the minimum average distance to the closest constellation point for each rotated input symbol, $\min(|e_0|^2, |e_1|^2, ..., |e_{B-1}|^2)$, and its two neighboring test phases, to interpolate intermediate phase rotations. Since input symbols rotated with these intermediate values are not calculated by the algorithm, a separate **compensation** component is required to remove the estimated phase noise from the input symbols. The total circuit complexity is, however, reduced as the number of test phases can be decreased at least four times. An added benefit is that the **compensation** component can be used as a slave unit when considering joint carrier phase recovery for, e.g., spectral super channels [13]. The **interpolation** component can be realized using the circuit shown in Fig. 2, where the differences $e_{b_{min}-1} - e_{b_{min}}$ and $e_{b_{min}+1} - e_{b_{min}}$ are calculated and used to index a look-up table (LUT). Implemented as a read-only memory, the LUT contains pre-calculated values of the least-significant bits of b_{comp}, which is concatenated with b_{min}, allowing for interpolation in steps of powers of two. The interpolation method allows us to reduce the number of test phases, and thus the power dissipation, of two of the largest components, **rotation** and **distance** in Fig. 1, by 75%.

The summation of the distances, $|d_{k,b}|^2$, in Algorithm 1 can be thought of as a sliding-window average, whose parallel implementation is very hardware intensive. In our case, this would require B L-input adders and $B \cdot (L - P)$ registers. To reduce circuit complexity, we use a two-stage block-wise average: An inner stage, where we add the P distances for each test phase, and, if $L > P$, an additional outer stage, where we store L/P succeeding results and calculate the total average. The result is that B P-input adders are needed, followed by one L/P-input adder and L/P registers, which is a significant reduction, especially for higher-order modulation formats which need a large B. An additional benefit is that we remove the requirement on parallelism in succeeding components, which further reduces complexity.

Algorithm 1: Blind phase search

input : X, input symbols
output : Z, output symbols
parameters: B, number of test phases
           L, averaging window size
Initialize test phases: $\phi_b = \frac{b}{B}\frac{\pi}{2}$, for $b = 0, 1, ..., B-1$
foreach *input sample, X_k* **do**
    foreach *test phase, b* **do**
         $X_{k,b} = X_k e^{-j\phi_b}$
         Rotate input symbol with test phase
         $|d_{k,b}|^2 = |X_{k,b} - \hat{X}_{k,b}|^2$.
         Calculate distance to closest constellation point $\hat{X}$
         $e_{k,b} = \sum_{i=-L/2}^{L/2} |d_{k+i,b}|^2$
         Reduce impact of AWGN by summing distance for L consecutive symbols
    end
    Find the test phase, b_{min}, resulting in smallest e
    $Z_k = X_{k,b_{min}}$
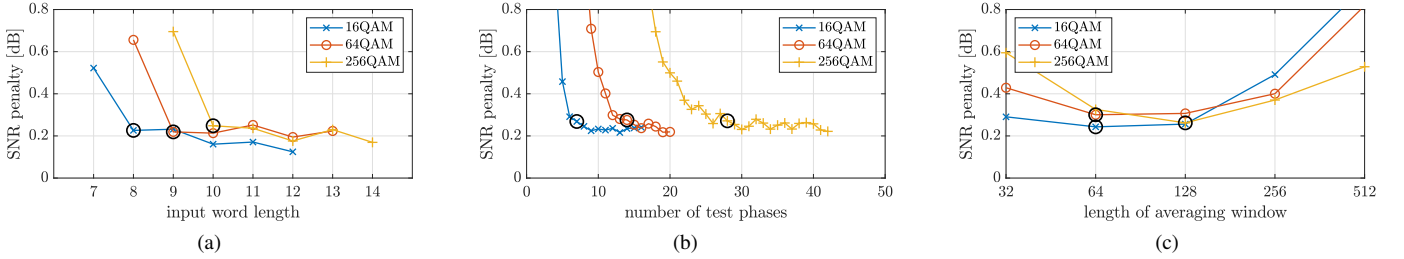   *Output input symbol rotated by b_{min}*
end

Fig. 4. Impact of parameter settings on the BPS SNR penalty compared to the theoretical minimum for (a) the input word length, (b) the number of test phases, and (c) the size of the averaging window, for linewidth symbol-duration products of $1 \cdot 10^{-5}$, $2 \cdot 10^{-6}$ and $4 \cdot 10^{-7}$ for 16, 64, and 256QAM, respectively. The circled points in each figure are kept constant in the other simulations.
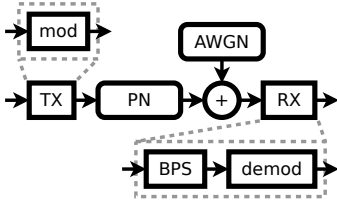

Fig. 3. System model used for simulations.

The resolution, i.e., signal word lengths, is kept as low as possible throughout the different components of the implementation, without losing information. Since we are only interested in the distance to the closest constellation point—not the actual point—the input signals can be mapped to the first quadrant at the input, eliminating the need for a sign bit at the succeeding stages. The distance to the closest constellation point is also at less than half of the maximum amplitude, allowing us to further reduce the number of bits after calculating d. This implementation approach means that the word length of all internal signals can be calculated directly from the input word length, N.

III. PARAMETER IMPACT ON SNR PENALTY

It is difficult to quantify the tradeoffs discussed in the previous section unless we can accurately evaluate alternate VLSI implementations in simulations of digital logic circuits, using hardware-description languages (HDLs). A system model of our simulation environment is shown in Fig. 3, including AWGN and phase noise (PN) generation; the latter modelled as a Wiener process [14]. We assume that all other linear impairments, e.g., chromatic dispersion and polarization-mode dispersion, are compensated for by other DSP units. In addition, we neglect the effect of non-linear impairments.

The ensuing simulations are performed as MATLAB-HDL co-simulations, where MATLAB is used to generate and modulate test data to emulate fiber transmission impairments. The symbols are then fed to a HDL-based software model of our BPS circuit implementation: This type of simulation, known as logic simulation, accounts for all implementation penalties, such as those due to a limited word length and to approximations used in the VLSI implementation. The output from the logic simulation is imported back to MATLAB for demodulation and calculation of the bit error rate (BER).

There are four main design parameters that can be adjusted for our BPS circuit implementations: The input word length (N), the number of test phases (B), and the averaging type and window size (L). How these are selected does not only impact the silicon area and power dissipation of the finished implementation, but also the algorithm performance. We will now investigate the CPR performance in terms of SNR penalty, i.e., how much the SNR must be increased compared to the theoretical limit to reach a BER of 10^{-2}, chosen as an approximation of a theoretical soft-decision FEC limit.

It is reasonable to assume that transmission systems using higher-order modulation formats would use lasers with a small linewidth, since the distance between the constellation points is smaller and the phase-noise sensitivity therefore is increased. In the following simulations, the linewidth symbol-period product was set to $1 \cdot 10^{-5}$ for 16QAM. The values for 64 and 256QAM were set to $2 \cdot 10^{-6}$ and $4 \cdot 10^{-7}$, respectively, since these values result in similar SNR penalties.

Fig. 4a shows the SNR penalty for varying word lengths of the input data, N, which in turn affects the word lengths used internally in the BPS circuit. When N is increased, the penalty becomes almost constant, indicating that a word length larger than 8, 9, and 10 bits for 16, 64, and 256QAM, respectively, would be unnecessarily large. For smaller N, the penalty is increased and for values less than 7, 8, and 9 bits, cycle slips become so frequent that it is impossible to plot any relevant BER curves. Cycle slips are caused by problems in the unwrapping of the phase and results in an erroneous rotation of the estimated phase by multiples of $\pi/2$. This problem affects many blind CPR methods, is very hard to recover from, and can potentially result in catastrophic transmission failure [15].

Analogous with the results shown in [8], increasing the number of test phases will decrease the SNR penalty of our circuit implementation, up until a limit when the improvement levels out, as shown in Fig. 4b. Thanks to the **interpolation** component, we use only a quarter of the number of test phases otherwise needed. The penalty curves start to level out at approximately 7, 14, and 28 test phases for 16, 64, and 256QAM, respectively, and these values are chosen for the rest of the simulations in this section. (Our values are, however, not directly comparable with [8] since that work assumed a lower BER of 10^{-3}.)

As shown in Fig. 5, the BER as a function of SNR for the two averaging types have similar performance. For higher
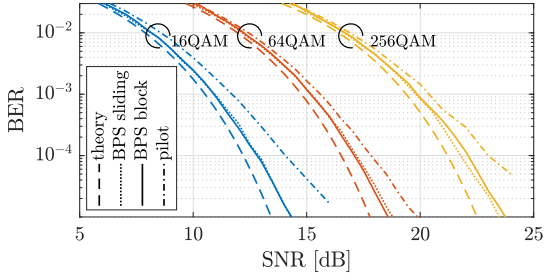
Fig. 7. Example of MCM transformation.

Fig. 5. BER curves for BPS, using both sliding-window and block averaging, and for our pilot-based CPR, using comparable parameter settings.
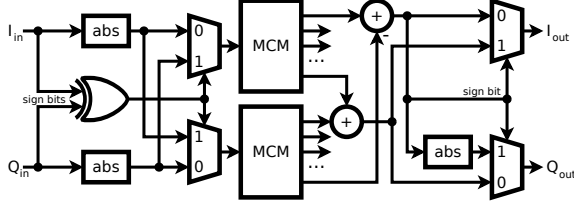

Fig. 6. Block diagram of the **rotation** component with mapping of the symbols to the first quadrant.

laser linewidths than used here, the sliding-window average implementation shows a slightly better performance, since it can handle fast phase fluctuations better. As we will show in Section VI, this however comes at the cost of a much higher power dissipation.

The SNR penalty as a function of the block-averaging window size, L, is shown in Fig. 4c, with minimum penalties at $L = 64$ for 16QAM and 64QAM, and at $L = 128$ for 256QAM. The position of this minimum is, however, strongly dependent on the amount of phase noise present in the received signal. A larger linewidth symbol-period product is equivalent with faster phase noise, which indicates that a shorter averaging window is needed to reduce the negative impact of phase noise on the CPR, to keep the output SNR penalty constant. However, a system with a smaller linewidth symbol-period product can use a larger L, which decreases the impact of AWGN on the CPR performance. Thus, a tradeoff exists between high-linewidth lasers and low SNRs, where BPS would have a problem handling both at the same time.

IV. POWER-EFFICIENT VLSI CIRCUIT DESIGN

This section will give an overview of the VLSI implementation of the BPS circuit components, emphasizing area usage and power dissipation aspects. The power dissipated by the BPS implementation is dominated by two components: The **rotation** with test phases and the calculation of the **distance** to the closest constellation point. This is because these two components operate on all B test phases in P parallel lanes, which means that PB circuit instances are needed. We will describe the components in the order in which they are connected, focusing on the ones which have the largest impact on the total power dissipation.

When the input symbols are received by the BPS unit in the DSP chain, the symbols use a signed IQ representation with a word length of N bits. Since BPS uses the *distance* to
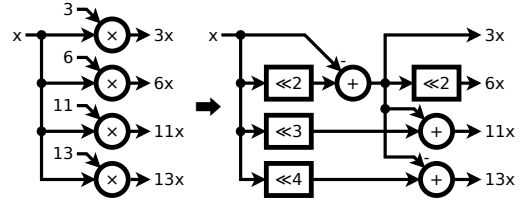
the closest constellation point—not the identity of the actual point—to perform carrier phase estimation, we can map the symbols to the first quadrant without affecting the output results, since the distances are the same. This translation is performed by the first part of the **rotation** component, shown in Fig. 6 for one parallel lane. The sign bits for the two input signals, I and Q, are XORed and used to control two output multiplexers. If the input symbol is in the first or third quadrant, the absolute values are fed directly to the output; if the symbols are in the second or fourth quadrant, the I and Q values are interchanged. The use of this mapping makes it possible for us to use an unsigned representation of the data in the following BPS components, reducing the word length to $N - 1$, which results in a lower total circuit complexity.

Rotation of the mapped input symbols is performed by multiplication with complex constants in polar form; $r = r_I + jr_Q = \sin\phi + j\cos\phi$, where ϕ is the test phase. Implementation of these multiplications would result in either three multipliers and four adders, or four multipliers and two adders per test phase. Since a multiplier is much more complex to implement than an adder, increasing both the power dissipation and the silicon area, the first option would be the preferred one. However, it would still result in a total of $3BP$ multiplier instances for the rotation.

One way of reducing the complexity of this multiple constant multiplication (MCM) problem is to simplify the multiplications to a shift-add network, illustrated in Fig. 7, where the intermediate results are shared between the test phases. ASIC design software, such as Cadence Genus [16], can perform this type of transformations for us. If we also take advantage of the fact that $\sin(\phi) = \cos(\pi/2 - \phi)$, the functionality can be realized using only two MCM operations as

$$I_{out} = I_{in}\cos(\phi) - Q_{in}\cos(\pi/2 - \phi),$$
$$Q_{out} = I_{in}\cos(\pi/2 - \phi) + Q_{in}\cos(\phi).$$

In the **rotation** component, the absolute value function and the two multiplexers are used to keep the output value in the first quadrant, thus removing the need to use a signed representation further downstream in the DSP chain. The multiplexers are controlled using the sign bit of the I part of the rotated symbol.

The distance from each rotated input symbol to its closest constellation point is calculated in the next BPS component, shown in Fig. 8, implemented using a number of comparators where the I and Q signals are compared separately to limits centered between the constellation points. The resulting row and column are used as index in a LUT to retrieve an IQ

Fig. 8. Block diagram of the **distance** component used to calculate the distance to the closest constellation point.

Fig. 9. Block diagram of the **compensation** component.

representation of the closest symbol. As the symbols are mapped to the first quadrant only, the number of comparators can be reduced by half and the size of the LUT can be reduced to a quarter of the values otherwise needed.

The original description of the BPS algorithm suggests that the square of the distance (d^2) should be calculated, to remove the need for a square-root operation. This method, however, introduces two VLSI circuit challenges, since two multiplications are needed and twice as many bits are used to store d^2 than to store d. Doubling the number of bits at the output from the distance calculation would have a large impact on the power dissipation and area usage of the components further downstream in the BPS chain. To calculate d without the use of multipliers, we use the $\alpha \max + \beta \min$ algorithm [17], which is an approximation of d as

$$d = |a + jb| = \sqrt{a^2 + b^2} \approx \alpha \max(a,b) + \beta \min(a,b).$$

With $\alpha = 1$ and $\beta = 1/4$, the result has a mean approximation error of 3.2%. The word length of the output, the distance d, can be set less than the input, since we know that the distance to the closest point is always smaller than the largest I and Q input. By also using a saturating adder in the final stage, the word length can be reduced even further without significantly affecting the CPR performance.

The **compensation** component, which is used to remove the estimated phase from the input symbols, handles multiple samples in parallel and one such parallel lane is shown in Fig. 9. The input data is delayed to synchronize with the output from the phase estimation part of the BPS algorithm using circular buffers. This type of buffers has two main advantages compared to a standard shift register. First, the switching activity can be reduced since we avoid updating all delay elements each clock cycle. Second, if clock gating is used, the clock signal can be turned off for the inactive delay elements. Each of these advantageous features results in substantial power dissipation reductions.

The estimated phase is used to index a LUT, which contains the rotation vector used to rotate the input symbols. The rotation itself is implemented as a complex multiplication, with

four multipliers and two adders. Since only one constant is used for each input symbol, this method is preferred over the previously described shift-add transformations and results in more power-efficient VLSI circuits than the three-multiplier implementation obtained from optimizations in our ASIC design software.

V. ASIC Evaluation Methodology

A hardware description of our BPS design was implemented and synthesized to a gate-level netlist using a 22-nm fully-depleted silicon-on-insulator (FD-SOI) CMOS cell library, characterized at the slow process corner, 0.72 V and 125°C. These settings are rather pessimistic, but would result in a good yield even at large process variations. We used Cadence Genus [16] for synthesis, assuming a clock rate of 1 GHz for all design variations. The implementation is, thus, parallelized in 32 lanes to reach a symbol rate of 32 GBaud.

Simulation of the gate-level netlist was carried out using MATLAB-HDL co-simulation, as described in Section III, and the switching statistics were back-annotated into Genus for power estimation. A library characterized at the typical process corner, 0.8 V and 85°C was used for power estimation, to be as close as possible to a normal usage case.

VI. Results

In this section, we will present results from VLSI implementations of BPS for a single polarization in a DSP ASIC, focusing on how the power dissipation varies with different parameter settings. A summary of these results is shown in Table I, for the parameter settings that were used in Section III.

The cell area of the designs scales with a doubling from 16QAM to 64QAM, and 2.5 times between 64QAM and 256QAM. When switching to a higher-order modulation format, all parameters need to be updated to reach a similar CPR performance, which affects the area of the design. The increased word length also affects the critical path of the VLSI circuit, forcing the synthesis tool to use larger cells to reach our timing goal at 1 GHz.

The power dissipation is dominated by dynamic power, which is related to the switching activity of the circuit. Even if we use implementation techniques to reduce switching, such as clock gating, the dynamic power is typically much larger than the static portion. Choosing a cell library with lower leakage could decrease the static power further, but would have a negative impact on the dynamic power since the cell drive strength, and thus the size and capacitance, would have to be increased to reach the timing goal. Scaling of the power dissipation to higher-order modulation formats follows a similar trend as the area scaling.

A. Word Length

Power dissipation for varying word lengths for our 64QAM BPS CPR implementation is shown in Fig. 10a. Since the word length of all internal signals is directly dependent on the input word length, all components are affected when the word length is increased. The dissipation of the **rotation** and **distance** components is affected more strongly, since these are
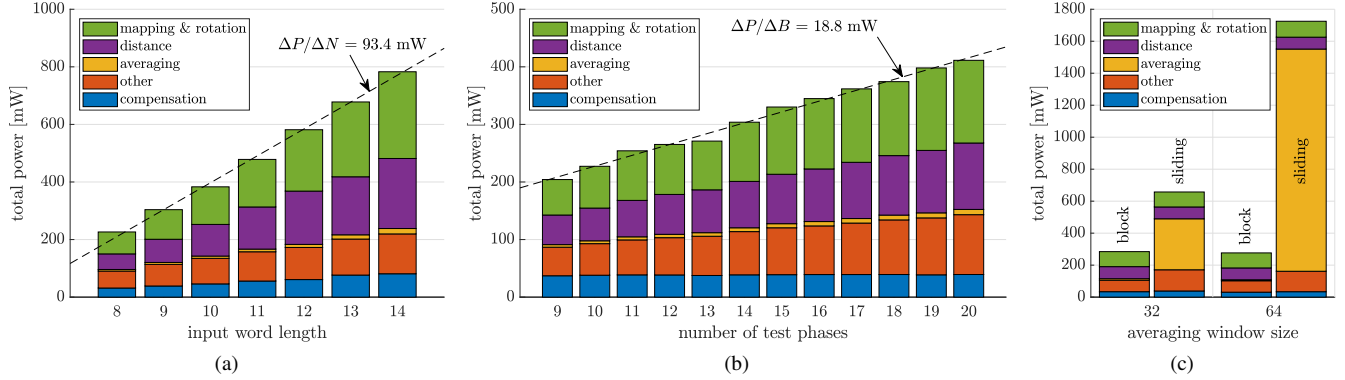
Fig. 10. Impact of parameter settings on the power dissipation of our 64QAM BPS implementation for different settings of (a) the input word length, (b) the number of test phases, and (c) the size of the averaging window.

TABLE I
SUMMARY OF IMPLEMENTATION RESULTS FOR A SINGLE POLARIZATION

Modulation format	Word length	Test phases	Averaging window	Area [mm²]	ΔA [mm²]		Power [mW]			ΔP [mW]		Area [mm²]	Total Power [mW]
					N	B	Leakage	Dynamic	Total	N	B		
16QAM	8	7	64	0.055	0.017	0.0049	14.1	128.3	142.4	40.2	15.6	0.024	48.1
64QAM	9	14	64	0.115	0.032	0.0063	31.0	272.8	303.8	93.4	18.8	0.030	69.5
256QAM	10	28	128	0.282	0.067	0.0089	80.4	709.3	789.7	211.7	27.7	0.036	86.3

The header above spans: "Blind Phase Search with Block Averaging" over columns Word length through ΔP [mW]; "Pilot-Based CPR" over Area and Total Power.

highly parallelized, perform calculations for all test phases, and are arithmetic intensive. For the same reasons, these two components also show the largest power dissipation, followed by the **others** group, in which all minor subcomponents and pipeline registers are included.

The relationships area vs. word length and power dissipation vs. word length both show a linear behavior, and we see the same type of behavior for all three modulation formats. The slope of the fitted line in Fig. 10a represents the increase in power dissipation per additional bit of word length ($\Delta P/\Delta N$) and Table I shows that this value is increased by a factor of 2.3 for each higher order modulation format. The values for area show similar behavior, but $\Delta A/\Delta N$ increases by closer to 2 for each higher order format.

B. Number of Test Phases

Fig. 10b shows the relationship between the number of test phases used and the power dissipation of the BPS circuit implementation. The number of test phases affects the complexity of a majority of the components shown in Fig. 1, even though the effect on the **interpolation** and **compensation** components is limited to extra entries in LUTs. The largest power increase with an increasing number of test phases can be seen for the highly parallelized components **rotation** and **distance**, as well as for the registers reported as a part of the **others** group.

Similarly to the word length, we see a linear increase of the total power dissipation with an increasing number of test phases, and $\Delta P/\Delta B = 18.8$ mW and $\Delta A/\Delta B = 0.0063$ mm² for our 64QAM implementation. These values scale with the selected modulation format and increase 1.2 times when switching from 16QAM to 64QAM, and 1.5 times when switching from 64QAM to 256QAM.

C. Averaging Window

For our block average implementation, the effect of a change in the averaging window size is negligible. Only the relatively small **average** component and the delay registers prior to the **compensation** component are affected. The differences are smaller than the differences in area and power caused by the heuristics of the synthesis tools that we can observe when performing a very minor design change.

To study the effect of using a block average as opposed to a sliding-window one, we also synthesized 64QAM implementations using the latter averaging method, with different window sizes. The distribution of the power dissipation between the components is shown in Fig. 10c, for designs using 14 test phases and 9-bit inputs. Compared to block averaging, the sliding-window designs show a total power dissipation more than five times higher, for $L = 64$, indicating that the latter method is impractical, especially since the SNR penalties are very similar, as we have shown in Fig. 5.

D. Pilot-Based Carrier Phase Recovery

As a reference, we developed a second CPR DSP unit using a pilot-symbol-aided (PAR) method, which uses known QPSK pilot symbols, time-division multiplexed with the data symbols, to recover the phase. A block diagram of our PAR circuit implementation is shown in Fig. 11, implemented in P parallel lanes. Here, the pilot is first demodulated and an average over L number of pilot symbols is calculated to reduce the impact of white noise. The average is then used to calculate the phase of the received pilots. This phase estimation is not necessarily performed every clock cycle, since the pilots can be inserted more sparsely than P samples apart. An **interpolation** component is used to interpolate the estimated
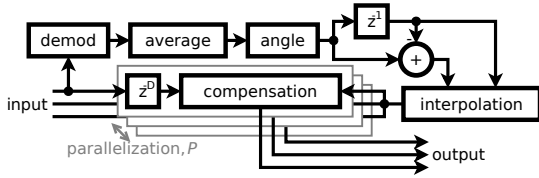
Fig. 11. Block diagram of pilot-based CPR implementation.

phase between the symbols. However, since phase changes are relatively slow compared to the symbol rate, we assume that all P parallel symbols have the same phase noise, reducing the number of LUTs needed for the conversion from angle to complex representation to one instead of P.

There are three main design parameters in our PAR implementation: the input word length, the CPR block length (C), and the average window size (L). The block length has the largest impact on CPR performance, both in terms of BER and spectral efficiency. Fig. 5 shows how the PAR algorithm compares to the BPS, with $C = 64$ (1.56% pilot overhead) and $L = 5$, where C is selected as high as possible while still being able to track the phase. L is chosen to minimize the effect of AWGN without affecting the phase tracking.

The estimated power dissipation of our PAR circuit implementation is shown in Table I. These values show that the PAR method is much more power efficient than BPS, albeit at lower spectral efficiency and at the cost of an SNR penalty, compared to BPS at a BER of 10^{-2}, of 0.4 dB for 16QAM and 0.2 dB for 64 and 256QAM. The scaling to higher order modulation formats is also much better for PAR, because the pilot symbols are QPSK, independent of the data modulation.

VII. CONCLUSION

We have introduced VLSI implementations of a blind phase search (BPS) carrier phase recovery (CPR) algorithm and discussed modifications of this algorithm to allow for efficient circuit implementation. Beginning with hardware description language (HDL) descriptions of the circuits, we performed netlist synthesis to a 22-nm process technology to accurately estimate power dissipation. This allowed us to explore different parameter settings and their effect on the resulting implementations, uncovering several tradeoffs: The choice of averaging type has a large impact on the power dissipation of the BPS design. Using a sliding-window average, as opposed to a block-wise calculation of the average, is deemed impractical, as the positive impact it has on CPR performance is negligible in comparison to the large increase in power dissipation that it causes. The three main design parameters are the input word length, the number of test phases, and the averaging window size. We have shown that the power dissipation depends linearly on the first two, while the effect of the last one is very small, assuming that block averaging is used. A relaxed laser linewidth constraint can be traded for an increased penalty; e.g., for 256QAM, a linewidth symbol-period of $2.5 \cdot 10^{-6}$ results in 0.4 dB higher SNR penalty than $4 \cdot 10^{-7}$. The only design parameter choice affected by such a change is the averaging window size, which has a very limited impact on the power dissipation.

Using a baudrate of 32 GBaud and a clock rate of 1 GHz, our BPS implementation dissipates 1.1 pJ/bit for 16QAM and 3.1 pJ/bit for 256QAM, for parameter settings selected as a good tradeoff between CPR performance and power dissipation. As a comparison, we have also included an implementation of pilot-based carrier phase recovery, with parameters set to reach a CPR performance level close to that of the BPS implementation, at a BER of 10^{-2}. The energy per bit for the pilot-based method is 0.38 pJ/bit for 16QAM and 0.34 pJ/bit for 256QAM. However, the spectral efficiency of these implementations is lower and a slightly higher SNR is needed to reach a BER of 10^{-2}.

REFERENCES

[1] C. Laperle and M. O'Sullivan, "Advances in high-speed DACs, ADCs, and DSP for optical coherent transceivers," *IEEE J. Lightw. Technol.*, vol. 32, no. 4, pp. 629–643, Feb. 2014.
[2] P. J. Winzer, D. T. Neilson, and A. R. Chraplyvy, "Fiber-optic transmission and networking: the previous 20 and the next 20 years," *Opt. Express*, vol. 26, no. 18, pp. 24190–24239, Sept. 2018.
[3] 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," *IEEE J. Lightw. Technol.*, vol. 32, no. 18, pp. 3093–3111, Sept. 2014.
[4] E. Maniloff, S. Gareau, and M. Moyer, "400G and beyond: Coherent evolution to high-capacity inter data center links," in *Opt. Fiber Commun. Conf. (OFC)*, 2019, p. M3H.4.
[5] A. Viterbi and A. Viterbi, "Nonlinear estimation of PSK-modulated carrier phase with application to burst digital transmission," *IEEE Trans. Inform. Theory*, vol. 29, no. 4, pp. 543–551, July 1983.
[6] I. Fatadin, D. Ives, and S. J. Savory, "Laser linewidth tolerance for 16-QAM coherent optical systems using QPSK partitioning," *IEEE Photon. Technol. Lett.*, vol. 22, no. 9, pp. 631–633, May 2010.
[7] P. Kam, "Maximum likelihood carrier phase recovery for linear suppressed-carrier digital data modulations," *IEEE Trans. Commun.*, vol. 34, no. 6, pp. 522–527, June 1986.
[8] T. Pfau, S. Hoffmann, and R. Noe, "Hardware-efficient coherent digital receiver concept with feedforward carrier recovery for M-QAM constellations," *IEEE J. Lightw. Technol.*, vol. 27, no. 8, pp. 989–999, Apr. 2009.
[9] A. Spalvieri and L. Barletta, "Pilot-aided carrier recovery in the presence of phase noise," *IEEE Trans. Commun.*, vol. 59, no. 7, pp. 1966–1974, July 2011.
[10] E. Ip and J. M. Kahn, "Feedforward carrier recovery for coherent optical communications," *IEEE J. Lightw. Technol.*, vol. 25, no. 9, pp. 2675–2692, Sep. 2007.
[11] H. Sun, K. Wu, S. Thomson, and Y. Wu, "Novel 16QAM carrier recovery based on blind phase search," in *Eur. Conf. Opt. Commun. (ECOC)*, Sept. 2014, p. Tu.1.3.4.
[12] E. Börjeson, C. Fougstedt, and P. Larsson-Edefors, "ASIC design exploration of phase recovery algorithms for M-QAM fiber-optical systems," in *Opt. Fiber Commun. Conf. (OFC)*, Feb. 2019, p. W3H.7.
[13] L. Lundberg, E. Börjeson, C. Fougstedt, M. Mazur, M. Karlsson, P. A. Andrekson, and P. Larsson-Edefors, "Power consumption savings through joint carrier recovery for spectral and spatial superchannels," in *Eur. Conf. Opt. Commun. (ECOC)*, Sept. 2018, p. We2.26.
[14] M. Magarini, A. Spalvieri, F. Vacondio, M. Bertolini, M. Pepe, and G. Gavioli, "Empirical modeling and simulation of phase noise in long-haul coherent optical transmission systems," *Opt. Express*, vol. 19, no. 23, pp. 22455–22461, Nov. 2011.
[15] E. Börjeson and P. Larsson-Edefors, "Cycle-slip rate analysis of blind phase search DSP circuit implementations," in *Opt. Fiber Commun. Conf. (OFC) 2020*, to appear.
[16] *Cadence® Genus®, v. 17.21*, Cadence Design Systems, Inc., 2018.
[17] R. G. Lyons, *Understanding digital signal processing*, 3rd ed. Prentice Hall PIR, 2011.