IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 26 December 2023, accepted 25 January 2024, date of publication 30 January 2024, date of current version 5 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3359713

RESEARCH ARTICLE

Real-Time Wide-Range Carrier Frequency Offset Compensation by Multi-Thread PADE for Fully Softwarized Access Networks

TAKAHIRO SUZUKI, (Member, IEEE), SANG-YUEP KIM, JUN-ICHI KANI, (Senior Member, IEEE), AND TOMOAKI YOSHIDA, (Member, IEEE)
NTT Access Network Service Systems Laboratories, NTT Corporation, Yokosuka-Shi, Kanagawa 239-0847, Japan
Corresponding author: Takahiro Suzuki (tkhr.suzuki@ntt.com)

ABSTRACT With the evolution of access systems by network function virtualization (NFV) and the software-defined network (SDN), further softwarization including transmission functions is required in order to fully utilize the advantages that softwarization offers to network operators in terms of more flexible and rapid service creation and migration. The softwarization of digital signal processing (DSP) is a prime target, but frequency offset compensation (FOC) has serial processing, which makes its softwarization difficult. In particular, the pre-decision-based angle differential estimator (PADE) of wide-range FOC needs long serial iterations since it obtains stable solutions by using small loop gain in the loop filter. This paper proposes a multiplication-based FOC implementation and a novel multi-thread PADE algorithm for real-time softwarization. The multiplication-based implementation enables parallel compensation of phase rotation by multiplying the estimated CFOs by sampling numbers for Viterbi-and-Viterbi (VV) method and PADE. In addition, our multi-thread PADE algorithm significantly reduces the iterations needed for FOC estimation; using multi-threading, it estimates multiple frequency offsets using the loop filters with large loop gain. This calculation needs relatively few iterations, but its outputs are very noisy, and final frequency offset estimates are obtained after removing the noise by clustering and averaging the results. The results of simulations and experiments, which use a general-purpose server with a graphic processing unit, show that our PADE algorithm is 1,308 times faster than the conventional implementation and 5-Gb/s real-time processing is achieved while maintaining the FOC performance.

INDEX TERMS SDN, PON, softwarization, DSP, frequency offset compensation.

I. INTRODUCTION

With the advent of sophisticated applications, autonomous robot control, augmented reality (AR), and tactile communications, the requirements placed on networks continue to become more stringent. Traditional central offices (COs), which are composed of dedicated hardware with application specific integrated circuits (ASICs), are barriers to the evolution needed to keep up with rapidly changing requirements. It is promising to use general-purpose processors in COs, as their performance is being improved by miniaturization in semiconductor fabrication processes. These processors offer agility and resource sharing among multiple services by combining network function virtualization (NFV) and software-defined network (SDN).

The application of NFV and SDN to optical access networks has been studied by major telecom operators. The AT&T and the Open Networking Foundation (ONF) advocated the approach of CO re-architected as a data center (CORD) [1]. To realize CORD for residential services, SDN-enabled broadband access (SEBA) has been developed and commercially deployed in some operators' networks [2]. The Broadband Forum (BBF) has also developed virtualized access systems and has considered co-configuration with

The associate editor coordinating the review of this manuscript and approving it for publication was Tianhua Xu.

TABLE 1. Summary of this study.

Aim	Challenge	Contribution
Softwarization of the upper-layer functions of optical access networks is actively progressing, and this paper targets expansion of the softwarization region to physical-layer processing including DSP in order to maximize the flexibility of the entire system. This approach enables agile development of entire functions and enables sharing of common resources with communication functions and edge applications.	In terms of frequency offset compensation, real-time softwarization of Viterbi-and-Viterbi method was achieved by block-wise multiplication-based FOC in the previous conference paper. To mitigate the requirement of wavelength control performance to LO, implementation of PADE, which compensates wide-range CFO, is desirable. However, its algorithm includes many iterations which makes softwarization difficult.	1). Proposal of a novel multi-thread PADE algorithm that reduces processing time by minimizing iteration number for DSP softwarization 2). Simulation results to analyze the iteration reduction effect of the proposed method and the estimated penalty of frequency offset compensation 3). Experimental evaluations to demonstrate real-time implementation of DSP software combining block-wise multiplication-based method, which was proposed in the previous conference paper, and the proposed multi-thread PADE

SEBA [3], [4]. The architecture implements abstracted functions of various optical line terminal (OLT) and optical network unit (ONU) specifications, and softwarized management and control functions even for different technologies [5]. The technologies are expected to bring agile development of access network services in commercial environments and lower total cost of ownership (TCO) to network operators. However, since the software region addressed in their studies is limited to upper-layer functions, it is difficult for access systems to flexibly change transmission functions and share resources between dedicated transmission system and other upper-layer functions.

For this background, media access control (MAC) and physical-layer (PHY) processing in the passive optical network (PON) have also been studied [6], [7]. For PHY processing, 10G-EPON processing has been demonstrated including forward error correction (FEC) [8], PON frame synchronization [9], and scrambler [10]. Additionally, given the trend to apply digital signal processing (DSP) in access networks [11], softwarization of DSP has been commenced. While some works softwarize DSP for applications such as data center interconnect [12], [13], 10 Gb/s DSP including carrier phase recovery (CPE), polarization demultiplexing, and decoding have been demonstrated running on general-purpose graphic processing units (GPUs) [14], [15]. While GPUs achieve high-speed processing by parallel processing on many cores, the challenge was serial processing of DSP, which cannot be parallelized. Thus, these studies achieved real-time softwarization by proposing algorithms that reduce serial processing. For frequency offset compensation (FOC), we realized real-time software implementation of the Viterbi-and-Viterbi (VV) method [16] by block-wise multiplication-based FOC [17]. However, its ability to compensate carrier frequency offset (CFO) is limited to [-Br/2m, Br/2m] for m-PSK signals where Br is symbol rate. To mitigate the requirement of wavelength control performance to local oscillator (LO), pre-decision-based angle differential estimator (PADE), which supports CFO of [-Br/2, Br/2], was proposed [18]. Although the softwarization of PADE is desirable, its algorithm has serial processing parts that require many iterations, making softwarization difficult even for GPUs. The large number of iterations are caused by the loop filter using a small loop gain to stabilize the solution.

If the entire suite of DSP functions is softwarized, similar to the trends in the adoption of general-purpose servers with GPU for signal processing in 5G radio access networks (RANs) [19], it would allow common general-purpose resources to implement both the communication functions and edge service functions, including augmented reality (AR) / virtual reality (VR), intelligent video analytics, and robotics tele-operations. The advantages include the ability to implement communication functions for specific purposes with small initial investment without developing specialized equipment, ease of upgrading to more advanced functions, and the ability to lend out central offices to users as computing resources when communication functions are not in use.

This paper extends our previous conference paper [17] by proposing a novel multi-thread PADE algorithm for softwarization and demonstrating its real-time performance. While we apply the block-wise multiplication-based method to PADE as was done in VV, our algorithm reduces processing time by utilizing multiple pre-estimated results of CFOs of multi-thread PADE with large loop gains in loop filters and few iterations. The results contain a lot of noise, which we remove by applying clustering and averaging multiple pre-estimated CFOs. Experiments show that our proposal is 1,308 times faster than the conventional serial implementation and can achieve real-time processing. Summary of this study is shown in Table 1.

This paper is organized as follows. Section *II* introduces the conventional FOC implementation method. Section *III* describes the proposed method including multiplication-based implementation and multi-thread PADE. Section *IV* shows simulation results, and Section *V*

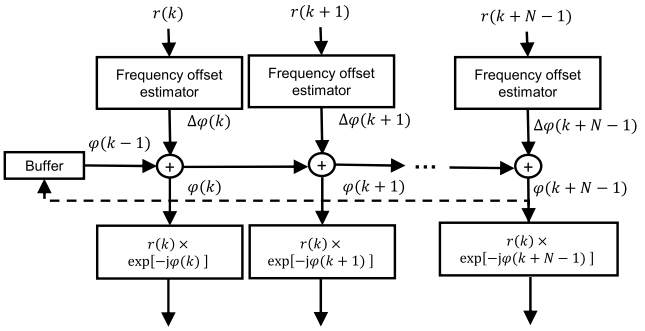


FIGURE 1. Conventional implementation of FOC.

demonstrates our real-time experimental evaluation. Finally, Section *VI* concludes this paper.

II. CONVENTIONAL METHODS

Before introducing our implementation method of FOC and extended method of PADE, this section describes the conventional implementation of FOC and the PADE algorithm.

Figure 1 illustrates a conventional typical FOC implementation. For k-th signal input $\mathbf{r}(k)$, one frequency offset estimator is executed, and N estimators can be parallelized for buffered N signals. Using calculated frequency offsets, $\Delta\varphi(k), \Delta\varphi(k+1), \cdots, \Delta\varphi(k+N-1)$, per signal, the phase rotation of the k-th symbol is compensated as follows;

$$\varphi(k) = \sum_{i=1}^{k} \Delta\varphi(i), \tag{1}$$

where the summation must be serially implemented. The summation of frequency offset, $\varphi(k+N-1)$, is buffered per N symbols to use it for subsequent buffered signals. While clock frequency of commercial FPGAs or ASICs is high, N values of 100 or less are sufficient for 10 Gb/s-class real-time systems, N must be larger for high-throughput software implementation because the short data transfers imposed by memory operations by both CPUs and GPUs reduce throughput. Thus, we need a method that reduces the FOC processing time with large N to take advantage of GPU resources like multiple threads.

Figure 2 shows the original PADE algorithm for wide-range frequency offset estimation. First, PADE calculates the phase error, which represents total amount of phase rotation with the CFOs at the current symbol. The phase error is obtained from the phase of $r(k)$ and pre-decision of the difference between the phase of $r(k)$ and the previously estimated phase error. The CFO is calculated from the estimated phase error and the previously estimated phase error. Given the estimated CFO has ambiguity of 2π, if the absolute value of the estimated CFO exceeds π, it is detected as an error. After that, a loop filter is utilized to increase solution stability and accuracy. As an example of implementation, $\Delta\varphi(k)$ is updated by

$$\Delta\varphi(k) \leftarrow \alpha\Delta\varphi(k) + (1-\alpha)\Delta\varphi(k-1), \tag{2}$$

where α is loop gain. Generally, a small value of α is utilized to stabilize the estimated results although this significantly slows convergence making FOC softwarization infeasible. If α is large, the stability of estimated results becomes low, but the estimation results lie near the solution $\pm n\pi/m$ with just a few iterations, where n is an integer. Since the conventional multiplication-based implementation of the PADE algorithm does not offer real-time performance, we propose here a novel PADE algorithm that supports multi-thread processing.

III. PROPOSED METHODS

First, this section introduces the proposed multiplication-based FOC implementation. Second, to reduce PADE iteration number and realize real-time software processing, it describes the proposed multi-thread PADE algorithm.

A. MULTIPLICATION-BASED IMPLEMENTATION

Figure 3 depicts the proposed multiplication-based FOC implementation that parallelizes the phase compensations of input signals by avoiding serial summation of CFOs. It calculates only one frequency offset $\Delta\varphi(k)$ per N symbols, and the phase is compensated with $\varphi(k-1)+i\Delta\varphi(k)$ for input signal $\mathbf{r}(k+i)$ on the i-th thread using N threads. $N\Delta\varphi(k)$ is buffered for calculating the subsequent N-symbol cycle frequency offset. This significantly reduces the need for serial processing shown in Eq. (1) and enables parallel processing of phase compensation after frequency offset calculation by the matrix,

$$\begin{pmatrix} r'(k) \\ r'(k+1) \\ \vdots \\ r'(k+N-1) \end{pmatrix} = \begin{pmatrix} exp[-j\varphi(k-1)+\Delta\varphi(k)] \\ exp[-j\varphi(k-1)+2\Delta\varphi(k)] \\ \vdots \\ exp[-j\varphi(k-1)+N\Delta\varphi(k)] \end{pmatrix} \circ \begin{pmatrix} r(k) \\ r(k+1) \\ \vdots \\ r(k+N-1) \end{pmatrix} \tag{3}$$

where $r'(k)$ is the compensated signal and $\circ$ is the Hadamard product.

B. MULTI-THREAD PADE

To reduce the PADE iteration number, we propose the novel multi-thread PADE shown in Fig. 4. The algorithm takes advantage of the feature that estimation results appear near the solution $\pm n\pi/m$ with just a few iterations, i.e., when α is large; pre-estimated results obtained with multiple estimators near the solution are selected and their average is determined as the final estimation result. It consists of four main parts: multi-thread frequency offset estimators, preprocessing for clustering, clustering, and frequency offset selection. The multi-thread frequency offset estimators parallelize the PADE algorithm for the input signals $r(k), r(k+1), \cdots, r(k+N-1)$. By using the pre-estimated results generated by

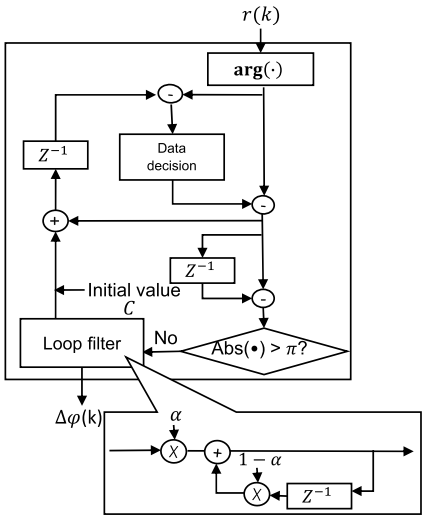


FIGURE 2. PADE.

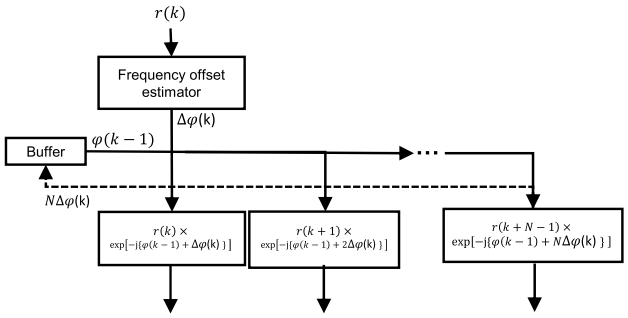


FIGURE 3. Proposed multiplication-based FOC implementation.

each thread, tolerance to noise is improved, even with few iterations. After that, the pre-estimated results are clustered, but clustering algorithms often fail to stabilize depending on initial values. Thus, we set a preprocessing clustering step to find the number and approximate means of the clusters of the estimated results and use them as initial values for algorithm. In concrete, a histogram of pre-estimated frequency offsets is constructed. The histogram $[h_0, h_1, \cdots, h_{M-1}]$ is calculated by

$$b_k = \left\lfloor M \frac{\Delta\varphi(k) + \pi}{2\pi} \right\rfloor, \tag{4}$$

$$h_{b_k} \leftarrow h_{b_k} + 1, \tag{5}$$

where b_k is the bin number of the k-th signal and M is the maximum bin number. The number of local maxima of the histogram is used as the number of clusters and the bins with local maxima are determined as the approximate means of each cluster. The differential histogram $[d_0, d_1, \cdots, d_{M-1}]$ is calculated by

$$d_i = h_i - h_{i-1}, \tag{6}$$

for the i-th bin. The number of clusters k_c and the approximate means of each cluster $V_{(\cdot)}$ are given by

$$V_{k_c} = \frac{2\pi i}{M} - \pi + \frac{\pi}{M} \ (if\ d_{i-1} < d_i < d_{i+1}), \tag{7}$$

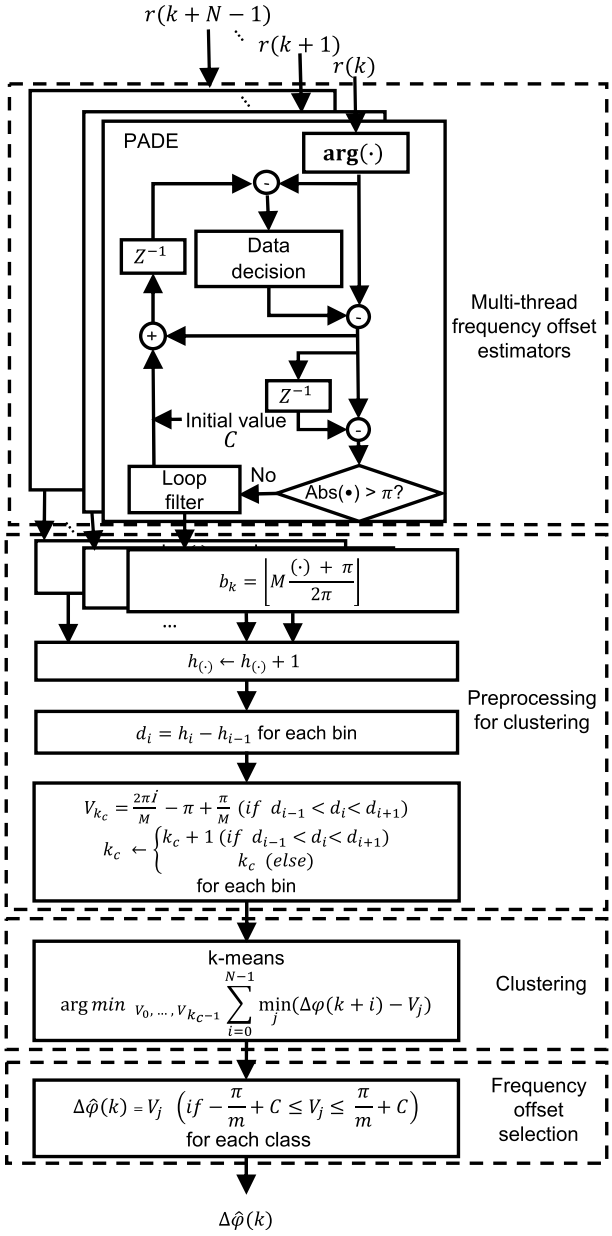


FIGURE 4. Proposed multi-thread PADE.

$$k_c \leftarrow \begin{cases} k_c + 1 \ (if\ d_{i-1} < d_i < d_{i+1}) \\ k_c \quad\quad (else) \end{cases}. \tag{8}$$

As the clustering algorithm, we utilize k-means. The number of clusters and mean of each cluster are taken as initial values and mean of each cluster is refined by the equation written as

$$arg\ min_{V_0,\cdots,V_{k_c-1}} \sum_{i=0}^{N-1} \min_j (\Delta\varphi(k+i) - V_j). \tag{9}$$

In frequency offset selection, the cluster near the solution is selected from among the multiple clusters. Given that the PADE algorithm can track frequency offset in the range of $[-\pi/m + C, \pi/m + C]$ during one convergence without

Algorithm 1 Multi-Thread PADE After Multi-Thread Frequency Offset Estimators

Input: $\Delta\varphi(k), \Delta\varphi(k+1), \cdots, \Delta\varphi(k+N-1)$
Output: $\Delta\hat{\varphi}(k)$
  while TRUE **do**
    for i=0 to M-1 **do**
      $h_i = 0$
    end for
    for i=0 to N-1 **do in parallel**
      $b_{k+i} = \left\lfloor M \frac{\Delta\varphi(k+i)+\pi}{2\pi} \right\rfloor$
    end for
    for i=0 to N-1 **do**
      $h_{b_{k+i}} = h_{b_{k+i}} + 1$
    end for
    for i=1 to M-1 **do**
      $d_i = h_i - h_{i-1}$
    end for
    $k_c = 0$
    for i=2 to M-2 **do**
      if $d_{i-1} < d_i < d_{i+1}$ **then**
        $V_{k_c} = \frac{2\pi i}{M} - \pi + \frac{\pi}{M}$
        $k_c = k_c + 1$
      end if
    end for
    for i=0 to L-1 **do**
      for j=0 to N-1 **do in parallel**
        assign $\Delta\varphi(k+j)$ to the number of cluster k_c which has the closest mean
      end for
      for j=0 to N-1 **do**
        calculate new mean of cluster and update $V_{(\cdot)}$
      end for
    end for
    for j=0 to k_c-1 **do**
      if $-\frac{\pi}{m} + C \le V_j \le \frac{\pi}{m} + C$ **then**
        $\Delta\hat{\varphi}(k) = V_j$
      end if
    end for
    $k = k + N$
    $C = \Delta\hat{\varphi}(k)$
  end while

ambiguity, where C is the initial value of frequency offset, the mean of the clusters in this range is chosen as the final solution.

The multi-thread frequency offset estimators first run the existing PADE algorithm on multi threads to generate estimated CFO, $\Delta\varphi(k), \Delta\varphi(k+1), \cdots, \Delta\varphi(k+N-1)$, as shown in section *II*. A complete algorithm that details the unique processing after the multi-thread frequency offset estimators is shown in Algorithm 1. This algorithm takes as input the estimated CFO calculated in each thread and outputs the final CFO $\Delta\hat{\varphi}(k)$ for N input signals. Processing is repeated in a while-loop for each N-length signal buffered in the server. In the preprocessing for the clustering part, after initializing each bin of the histogram, h_i, and the number of clusters, k_c, the appearance frequency of bin, h_i, the differentiation of the histogram, d_i, the number of clusters, k_c, and the initial values of the average value of each cluster, V_i, are calculated. The calculations needed for each bin of the histogram can be parallelized. The clustering part repeats the assignment of estimated CFO to the cluster number, which has the closest mean, and calculates the new mean of the cluster L times so that the cluster mean converges. Finally, only the mean of cluster in the specified range is selected as final $\Delta\hat{\varphi}(k)$ in frequency offset selection.

While this algorithm can be applied to modulation formats for m-PSK signals, in order to apply it to quadrature amplitude modulation (QAM), it is necessary to extend the algorithm by utilizing amplitude information.

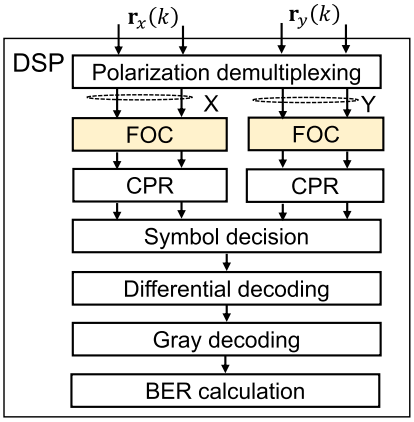


FIGURE 5. Simulated function blocks.

IV. SIMULATION

To find optimal parameters for stable operation and evaluate the performance of the conventional methods and the proposed methods, Monte Carlo (MC) bit error rate (BER) simulations are carried out. First, the optimal parameters of conventional VV and PADE are found to compare computation time. After that, the parameters of the proposed multi-thread PADE are also decided. Finally, their FOC performances are shown.

A. SIMULATION MODEL

The MC BER simulation models phase rotation by frequency offset f_o. If we let $\mathbf{s}_x(k)$, $\mathbf{s}_y(k)$ and $\theta_x(k)$, $\theta_y(k)$ be the amplitude and phase of a modulated pseudo-random binary sequence (PRBS) after Gray encoding and differential encoding at symbol rate f_s, the signals $\mathbf{r}_x(k)$, $\mathbf{r}_y(k)$ input to the simulated DSP blocks are described as

$$\begin{pmatrix} \mathbf{r}_x(k) \\ \mathbf{r}_y(k) \end{pmatrix} = \mathbf{J} \begin{pmatrix} \mathbf{s}_x(k) e^{j\{\theta_x(k)+\theta_n(k)\}} \\ \mathbf{s}_y(k) e^{j\{\theta_y(k)+\theta_n(k)\}} \end{pmatrix} + \begin{pmatrix} n(k) \\ n(k) \end{pmatrix}, \quad (10)$$

$$\theta_n(k+1) = \theta_n(k) + 2\pi \frac{f_o}{f_s}, \quad (11)$$

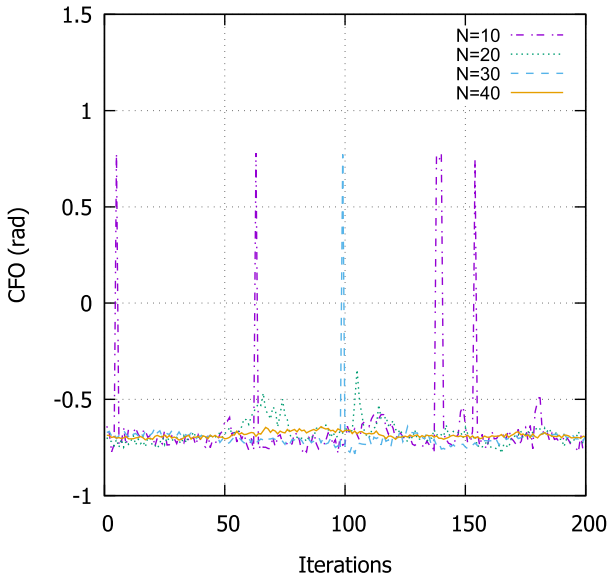


FIGURE 6. Estimated CFOs by VV versus N.

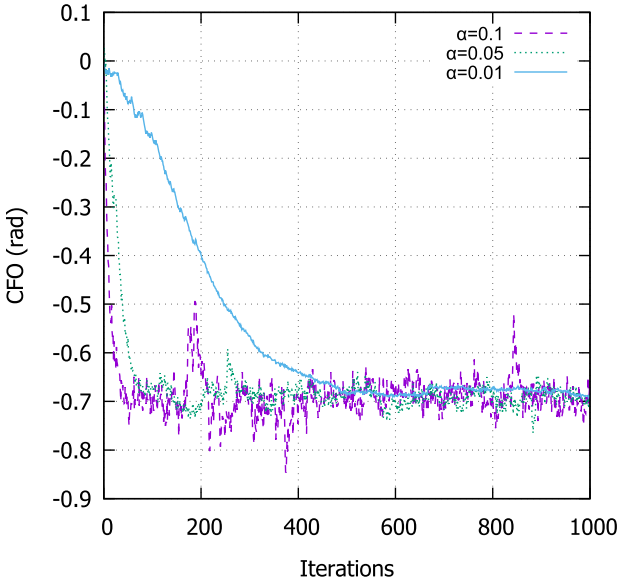


FIGURE 7. Estimated CFOs by PADE versus α.

$$\mathbf{J} = \begin{pmatrix} \cos\beta & \sin\beta \\ -\sin\beta & \cos\beta \end{pmatrix}, \tag{12}$$

where β is a constant amount of polarization rotation, $n(k)$ is noise, and $\theta_n(k)$ is the phase noise.

The encoded dual-polarization-quadrature phase-shift keying (DP-QPSK) signals, $\mathbf{r}_x(k)$, $\mathbf{r}_y(k)$, are input to the simulated function blocks as shown in Fig. 5. After polarization demultiplexing is performed, the FOC is performed. After that, the simulated signals are demodulated by carrier phase recovery (CPR) and symbol decision. Finally, BER calculation is carried out for the signals decoded with differential decoding and Gray decoding. In this simulation, we utilized the parameter of $\beta = \pi/8$ for polarization rotation. The number of iterations L of k-means in Algorithm 1 was set to 2, and signal-noise ratio (SNR) was set to 11.8 dB assuming BER=10^{-3}.

B. SIMULATION RESULTS

To confirm the parameter that achieves stable FOC performance, first, we simulated CFO estimation by the conventional VV method for several iteration numbers as shown in Fig. 6. In this figure, N represents the number of samples to be averaged to the fourth power. We set $2\pi f_o/f_s$ to -0.7539 (rad) due to the value that is near the limit of compensable range. If we define convergence as occurring when the estimated value is within 10 % of the target value, $N = 40$ is the appropriate value. The value will be utilized below for comparing the processing time.

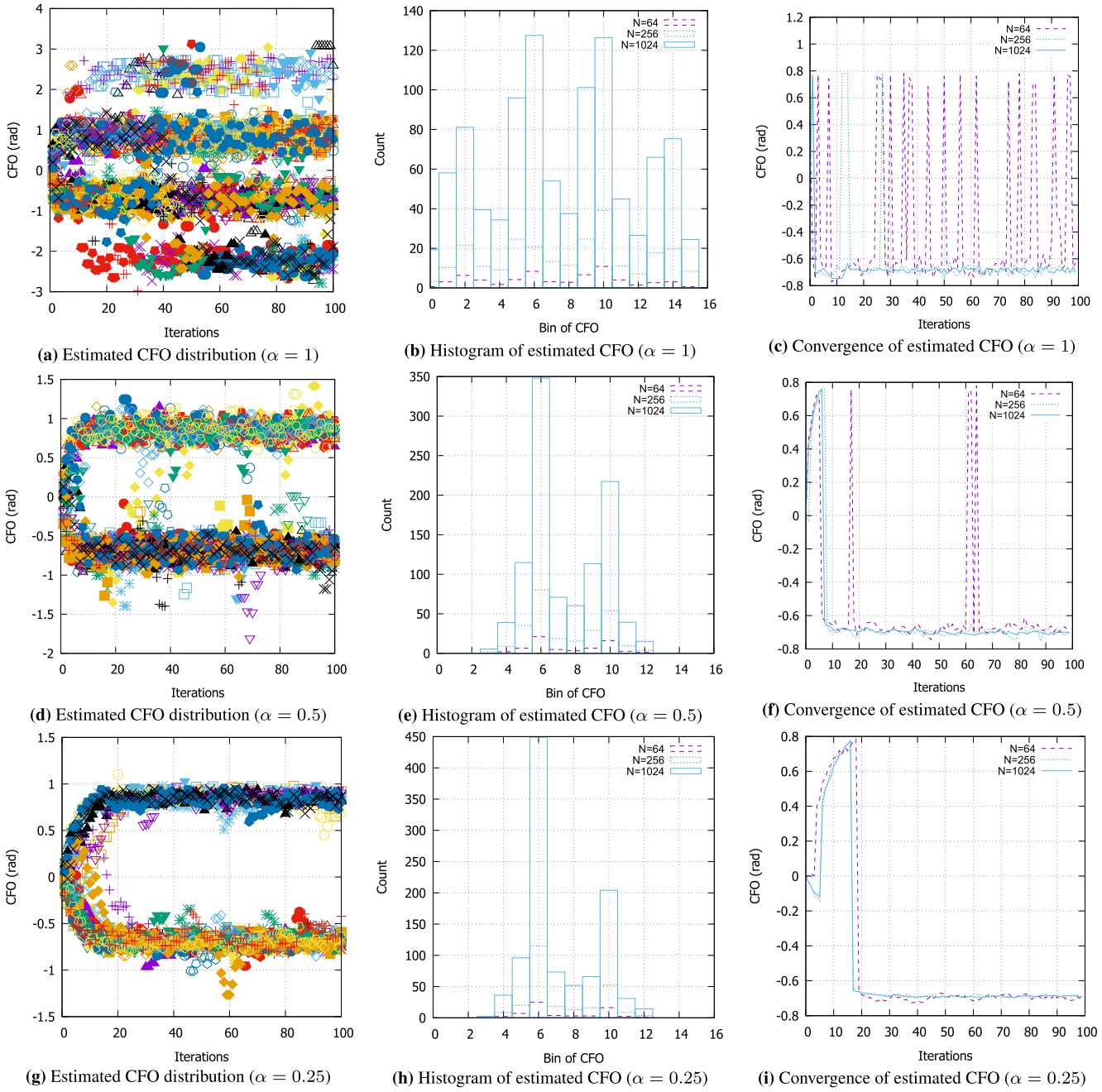
Figure 7 shows the estimated CFOs of the conventional PADE for various iteration numbers to determine the optimal parameter α that matches the performance for comparison of the processing time. From the figure, while the estimated values fluctuate with $\alpha = 0.1, 0.05$ even if the number of iterations is large, in the case of $\alpha = 0, 01$, convergence is achieved. However, the iteration number to convergence is 531, which greatly lengthens the serial processing operation; it is this part that is reduced by our algorithm.

Next, to determine the optimal parameters of the proposed multi-thread PADE, Figure 8 shows the output results of each method for each parameter α in the loop filter and the number of threads, N. Figures 8 (a) (d) (g) show the estimated CFO distribution yielded by the multi-thread frequency offset estimators in Fig 4. Although the target frequency offset is -0.7539 (rad), the estimated values are output in $\pm\pi/4$ cycles and estimated values appear near the solution with just a few iterations. Figures 8 (b) (e) (h) show histograms constructed in the clustering preprocessing step of Fig 4. They correctly represent the distribution of estimated CFO values and have graph shapes that make it possible to detect local maxima. Figures 8 (c) (f) (i) show the estimated CFO versus iteration number. The iteration number for convergence in Fig. (f) is 12, a significant reduction compared with the conventional PADE. We selected $N = 1024$ and $\alpha = 0.5$ as the optimal parameters considering short convergence.

Figure 9 shows the FOC performance results of each method when penalty was measured at the baseline of BER=10^{-3}. Three methods, the conventional PADE, the multiplication-based PADE, and the multi-thread PADE with the multiplication-based implementation, realize almost the same performance. This result shows that our proposed algorithm has no performance penalty.

V. EXPERIMENTAL EVALUATION

Our proposals were implemented on a server with a GPU, and real-time performance was evaluated using a coherent 5-Gb/s DP-QPSK system. We evaluated processing time to confirm real-time execution, CFO performance, and BER performance.



(a) Estimated CFO distribution ($\alpha = 1$)
(b) Histogram of estimated CFO ($\alpha = 1$)
(c) Convergence of estimated CFO ($\alpha = 1$)
(d) Estimated CFO distribution ($\alpha = 0.5$)
(e) Histogram of estimated CFO ($\alpha = 0.5$)
(f) Convergence of estimated CFO ($\alpha = 0.5$)
(g) Estimated CFO distribution ($\alpha = 0.25$)
(h) Histogram of estimated CFO ($\alpha = 0.25$)
(i) Convergence of estimated CFO ($\alpha = 0.25$)

FIGURE 8. **(a) (d) (g) CFO distribution estimated by PADE, (b) (e) (h) histogram of estimated CFO, and (c) (f) (i) convergence of estimated CFO versus α and N for large values of α.**

A. EXPERIMENTAL SETUP

Figure 10 shows the experimental setup. The server had two CPUs (Intel Broadwell Xeon E5-2699v4, 2.20 GHz, 22 cores, 44 threads) and a GPU (NVIDIA Tesla A100), which had 6912 CUDA cores. The GPU board included 40 GB of high bandwidth memory (HBM) 2E memory was linked to the CPU and interface (IF) card via PCIe gen3 $\times$ 16. The periodic 134 Mbyte data input is stored in this memory and softwarized DSP was iterated on the GPU. After CPU transferred the input ADC signals to GPU, the GPU kernel performed the coherent receiver DSP shown in Fig. 5. The software was written in CUDA, C++, and C languages according to the pseudocode shown in Algorithm 1. The source code was developed in-house and is not made public. The IF card was equipped with an FPGA (Xilinx DK-V7-VC709-G) connected to four 1.25-GSample/s ADC modules with 8-bit resolution (4DSP FMC125). The sampling point in the ADC is manually optimized without oversampling because no timing recovery function was implemented. A distributed feedback (DFB)

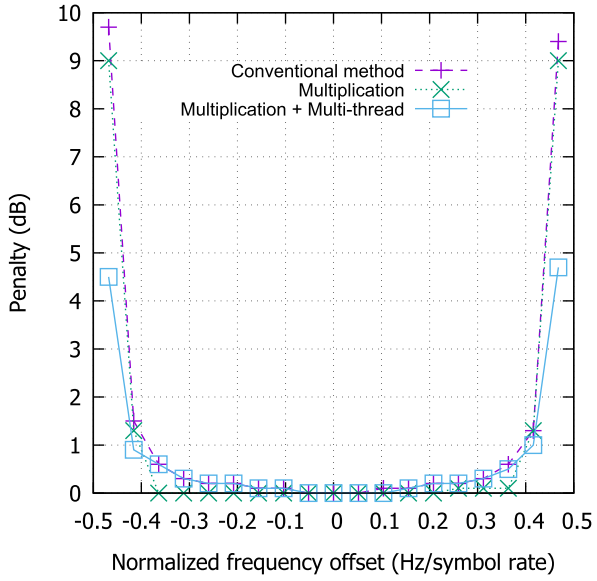


FIGURE 9. Penalty for normalized frequency offsets.

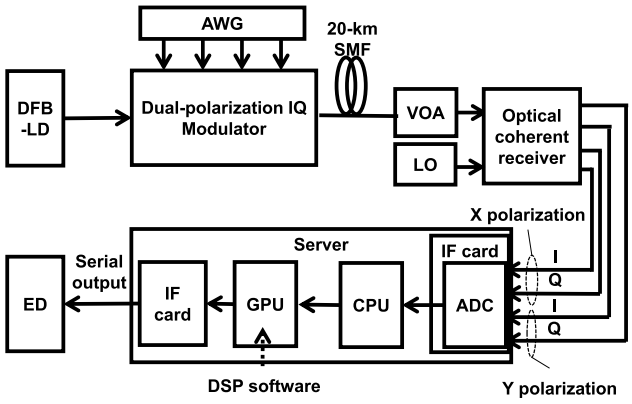


FIGURE 10. Experimental setup.

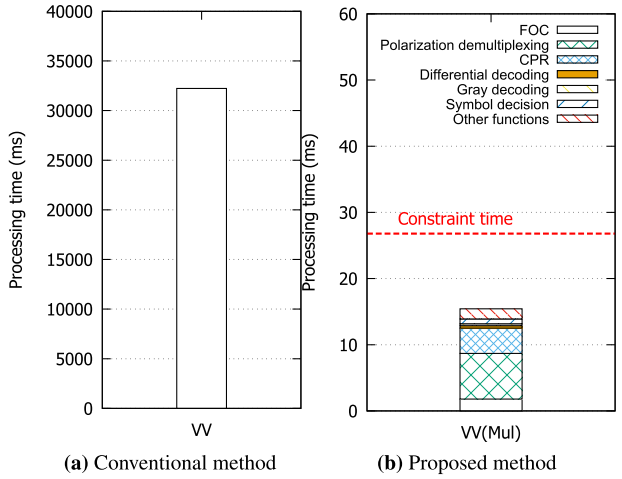


FIGURE 11. Processing time of VV. (Mul: multiplication-based implementation.)

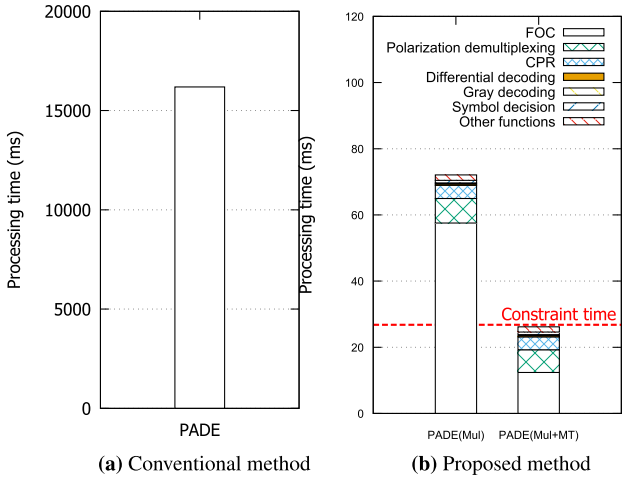


FIGURE 12. Processing time of PADE. (Mul: multiplication-based implementation, MT: multi-thread algorithm.)

laser diode (LD) with 80-kHz linewidth was driven as a transmitter at approximately 1553 nm. 1.25-Gsymbol/s DP-QPSK signals with launched powers of −4.45 dBm generated by an arbitrary waveform generator (AWG) programmed with the signals of differential and Gray encoding of each pattern (X polarization: PRBS 23, Y polarization: PRBS 17), were modulated by a LiNbO3 optical IQ modulator. The power of the received optical signal passed over a 20-km single mode fiber (SMF) was adjusted by a variable optical attenuator (VOA) and input to the coherent optical receiver (Fujitsu FIM24723EB). The local oscillator (LO) was the same type of LD used in the transmitter; its power was 14 dBm. The 5-Gb/s demodulated output of the server was serially input to an error detector (ED) to confirm successful completion of DSP. In terms of software evaluation, the average values of over 1,000 function calls are plotted. For processing time, given that in our experiments the measured standard deviation was about 3.0 μs, the standard error is calculated to be 94 ns by 3 μs $/\sqrt{1000}$, which is sufficiently small for measured processing times of several tens of millisecond and indicates the high reliability of the average value.

B. EXPERIMENTAL RESULTS

First, to evaluate the effects of the proposed method, the processing time for 134-Mbyte buffered data was measured as shown in Fig. 11. In order to execute 5 Gb/s demodulation for a 40 Gb/s signal input (i.e. 1.25-GS/s sampling rate, 4 channels and 8 bit vertical resolution), software processing of DSP must be completed within 26.8 ms. While the processing time of the conventional VV method is huge as shown in Fig. 11 (a), the proposed multiplication-based implementation has significantly reduced processing time such that it satisfies the constraint time for real-time processing as shown in Fig. 11 (b). This is because the conventional method estimates the CFO using serial processing as shown in Fig. 1, whereas the proposed method estimates the CFO in parallel by utilizing multiple cores of the GPU using the multiplication-based method as shown in Fig. 3. In particular,

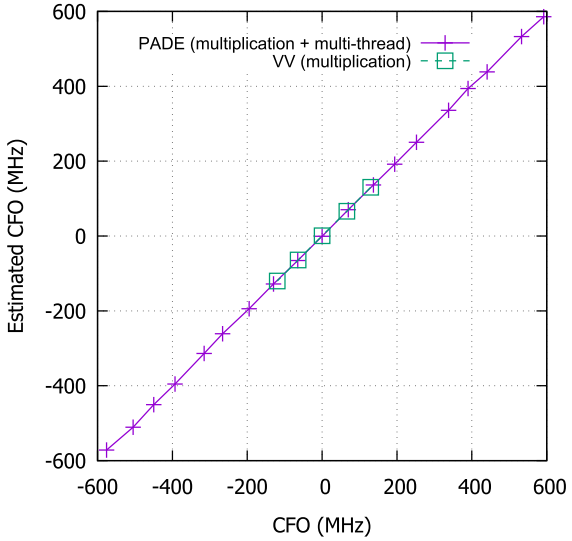


FIGURE 13. Estimated CFOs versus actual CFOs.

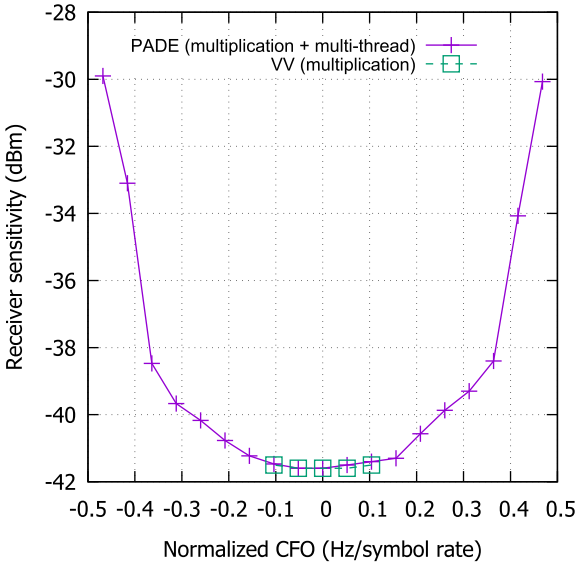


FIGURE 14. Receiver sensitivity versus normalized CFO.

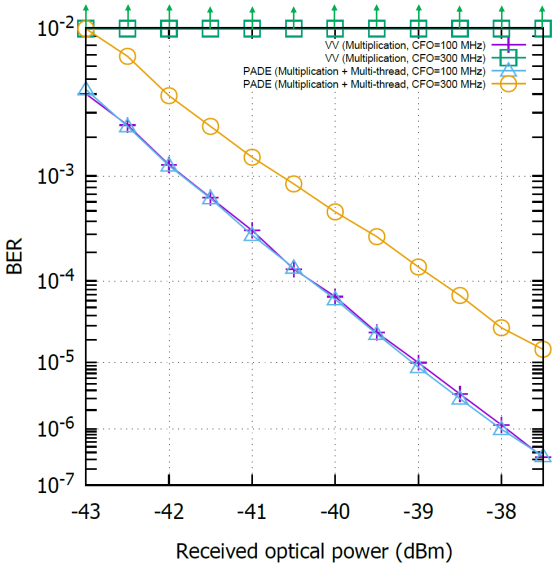


FIGURE 15. BER performance.

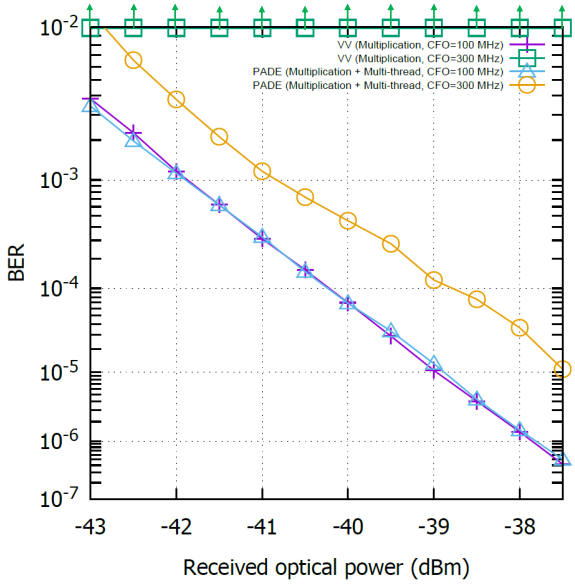
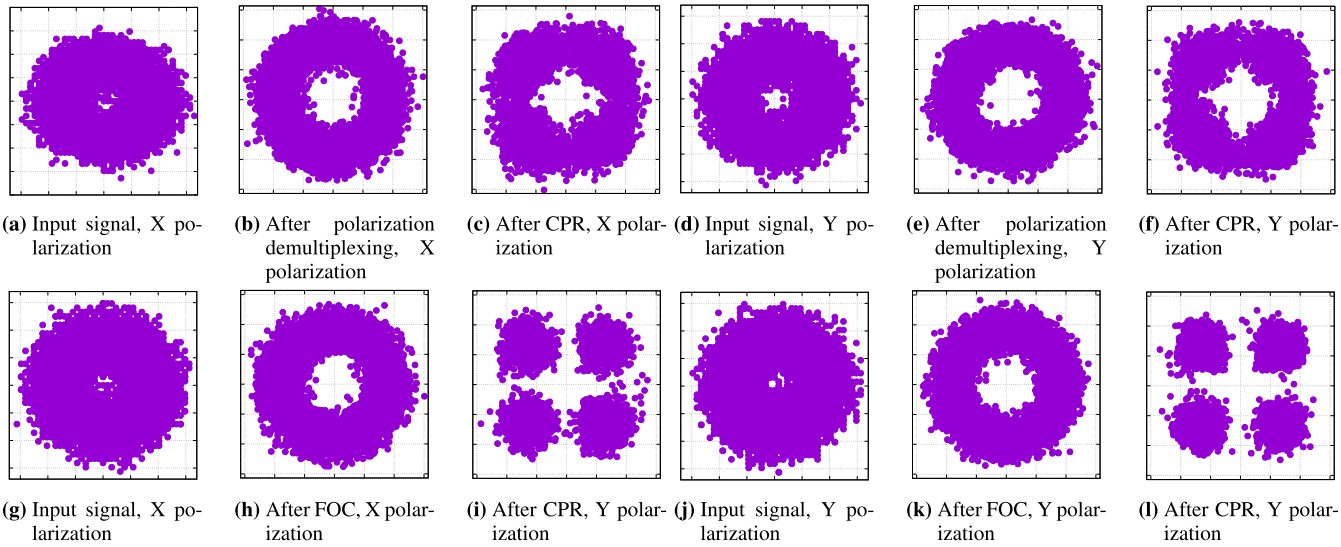


FIGURE 16. BER performance with 20-km fiber.

$exp[-j\varphi(k-1)+(i+1)\Delta\varphi(k)]\cdot r(k+i)$ in Eq. (3) is executed in parallel on N threads (i.e. $i=0,\cdots,N-1$), where 131,072 was utilized as N. Figure 12 shows the processing time of PADE until estimated CFO converges. The processing times of PADE and multiplication-based implementation of PADE significantly exceed the constraint time as shown in Fig. 12 (a) and (b). Figure 12 (b) shows that our multi-thread PADE with multiplication-based implementation is 1,308-times faster than the conventional method. The iteration number to convergence of the conventional method is 531 while that of the proposed method is 12 as shown in Figs. 7 and 8 (f), and this reduction in iteration number leads to a significant reduction in processing time, which allows general-purpose servers to satisfy the constraint time of 5 Gb/s real-time processing. These results indicate that our proposed implementation and algorithm successfully reduce the processing time so that real-time software processing is achieved.

Next, FOC performance was evaluated. Figure 13 shows estimation results of the proposed methods when the BER is 10^{-3}. Given that symbol rate is 1.25 Gsymbol/s, the theoretical CFO tolerance of the VV method is approximately $\pm$156 MHz, whereas that of PADE is $\pm$625 MHz. In the figure, the estimated CFOs are close to the actual CFOs in the range of the theoretical values examined. Figure 14 shows the receiver sensitivity versus normalized CFOs when the BER is 10^{-3}. In terms of normalized CFOs, the theoretical tolerance of VV method is approximately $\pm$0.125 (Hz/symbol rate), whereas that of PADE is $\pm$0.5 (Hz/symbol rate) in the absence of noise. This result shows that our proposed PADE is more robust against



(a) Input signal, X polarization **(b)** After polarization demultiplexing, X polarization **(c)** After CPR, X polarization **(d)** Input signal, Y polarization **(e)** After polarization demultiplexing, Y polarization **(f)** After CPR, Y polarization

(g) Input signal, X polarization **(h)** After FOC, X polarization **(i)** After CPR, Y polarization **(j)** Input signal, Y polarization **(k)** After FOC, Y polarization **(l)** After CPR, Y polarization

FIGURE 17. **Constellation diagrams for (a)-(f) demodulation without FOC and (g)-(l) demodulation with proposed FOC for 300-MHz CFO.**

CFO fluctuations rather the VV implementation and that our proposals offer real-time software implementation of wide-range FOC.

Finally, BER performance and constellations were measured. Figures 15 and 16 show the measured BER of the proposed methods for back-to-back (BTB) and 20-km transmission, respectively. The BER is plotted against the received optical power for CFOs of 100 MHz and 300 MHz. Assuming the target BER is 10^{-3}, to suit the FEC utilized in access systems [20], the measured receiver sensitivity was -41.5 dBm for CFO of 100 MHz. In addition, the proposed PADE algorithm can compensate 300-MHz CFO, unlike the VV method. This result shows that the proposed PADE implemented on a practical server successfully achieved real-time processing. Data captured in GPU memory to assess the operations of the proposed FOC are shown in Fig. 17. Diagrams (a)-(f) show the constellations without FOC, while diagrams (g)-(l) show that with proposed FOC at BER$=10^{-3}$ and 300-MHz CFO. This result also shows that the proposed algorithm correctly compensated the CFOs of the input signal and realized demodulation.

As shown in Fig. 12, although throughput of PADE with the conventional software implementation method is calculated to be about 8.27 Mb/s as indicated by the processing time of 16.2 s for 134 Mbytes, our improved method achieved 5-Gb/s real-time DSP. This confirms the feasibility of applying our method to 5 Gb/s-class optical access interfaces targeting long reach or high-splitting ratio use cases in the current situation, whereas the conventional implementation is limited to 8 Mb/s interfaces. Moreover, with further performance improvements, it also targets the introduction of 100 Gb/s-class high-speed PON systems, an area of great research interest, by 2030. Given that the performance of NVIDIA GPU products is rising by about 29 % every year due to progress of semiconductor manufacturing technology, and considering that the GPU we used was made in 2020, we expect a roughly 12-fold improvement in symbol rate by 2030. Additionally, higher order modulation schemes than the current QPSK signaling can be considered since throughput can be enhanced $m/2$ times by using 2^m QAM.

VI. CONCLUSION

With the promotion of virtualized OLTs, DSP softwarization has become an urgent goal to enhance the flexibility and agility of access networks. FOC has been a barrier to full softwarization since it demands lengthy serial processing; PADE needs many serial iterations to estimate CFOs. This paper proposed a multiplication-based FOC implementation and a novel multi-thread PADE algorithm for real-time GPU implementation. Multi-threads are utilized for frequency compensation and frequency estimation. Our multi-thread PADE calculates multiple pre-estimated CFOs in just a few iterations and eliminates noise by clustering and averaging. MC simulation results showed that our PADE algorithm had no FOC performance penalty. Measurements on a 5-Gb/s DP-QPSK optical system showed that our implementation achieved real-time processing of continuous signal inputs by reducing the processing time by 1,308 times while attaining the desired performance. The conducted experiments confirm 5-Gb/s real-time DSP softwarization, including polarization demultiplexing, FOC, CPR, symbol decision, differential decoding, and Gray decoding. This wide-range FOC has a throughput comparable to hardware implementations, and it yielded by our computationally-efficient method well suits the full softwarization of access networks.

REFERENCES

[1] L. Peterson, A. Al-Shabibi, T. Anshutz, S. Baker, A. Bavier, S. Das, J. Hart, G. Palukar, and W. Snow, "Central office re-architected as a data center," *IEEE Commun. Mag.*, vol. 54, no. 10, pp. 96–101, Oct. 2016.
[2] S. Das, "From CORD to SDN enabled broadband access (SEBA) [invited tutorial]," *J. Opt. Commun. Netw.*, vol. 13, no. 1, pp. A88–A99, Jan. 2021.

[3] *Broadband Forum (BBF) TR-384 Cloud Central Office Reference Architectural Framework*, Broadband Forum (BBF), USA, 2018.

[4] (2022). *Broadband Forum CloudCO Demo*. [Online]. Available: https://www.broadband-forum.org/broadband-forum-cloudco-demo-2022

[5] T. Suzuki, Y. Koyasako, K. Nishimoto, K. Asaka, T. Yamada, J.-I. Kani, T. Shimada, and T. Yoshida, "Demonstration of IEEE PON abstraction for SDN enabled broadband access (SEBA)," *J. Lightw. Technol.*, vol. 39, no. 20, pp. 6434–6442, Oct. 15, 2021.

[6] M. Ruffini, A. Ahmad, S. Zeb, N. Afraz, and F. Slyne, "Virtual DBA: Virtualizing passive optical networks to enable multi-service operation in true multi-tenant environments," *J. Opt. Commun. Netw.*, vol. 12, no. 4, pp. B63–B73, Apr. 2020.

[7] T. Suzuki, S.-Y. Kim, J.-I. Kani, and J. Terada, "Demonstration of fully softwarized 10G-EPON PHY processing on a general-purpose server for flexible access systems," *J. Lightw. Technol.*, vol. 38, no. 4, pp. 777–783, Feb. 13, 2020.

[8] T. Suzuki, S.-Y. Kim, J.-I. Kani, T. Hanawa, K.-I. Suzuki, and A. Otaka, "Demonstration of 10-Gbps real-time Reed–Solomon decoding using GPU direct transfer and kernel scheduling for flexible access systems," *J. Lightw. Technol.*, vol. 36, no. 10, pp. 1875–1881, May 15, 2018.

[9] T. Suzuki, S.-Y. Kim, J.-I. Kani, A. Otaka, and T. Hanawa, "10-Gb/s software implementation of burst-frame synchronization using array-access bitshift and dual-stage detection for flexible access systems," *J. Lightw. Technol.*, vol. 36, no. 23, pp. 5656–5662, Dec. 12, 2018.

[10] T. Suzuki, S.-Y. Kim, J.-I. Kani, and J. Terada, "Software implementation of 10G-EPON downstream physical-layer processing adopting CPU-GPU cooperative computing for flexible access systems," *IEEE Access*, vol. 7, pp. 33888–33897, 2019.

[11] A. Teixeira, D. Lavery, E. Ciaramella, L. Schmalen, N. Iiyama, R. M. Ferreira, and S. Randel, "DSP enabled optical detection techniques for PON," *J. Lightw. Technol.*, vol. 38, no. 3, pp. 684–695, Feb. 1, 2020.

[12] S. van der Heide, R. S. Luis, B. J. Puttnam, G. Rademacher, T. Koonen, S. Shinada, Y. Awaji, H. Furukawa, and C. Okonkwo, "Field trial of a flexible real-time software-defined GPU-based optical receiver," *J. Lightw. Technol.*, vol. 39, no. 8, pp. 2358–2367, Apr. 15, 2021.

[13] S. van der Heide, R. S. Luis, B. J. Puttnam, G. Rademacher, T. Koonen, S. Shinada, Y. Awaji, H. Furukawa, and C. Okonkwo, "Real-time 10,000 km straight-line transmission using a software-defined GPU-based receiver," *IEEE Photon. Technol. Lett.*, vol. 33, no. 24, pp. 1519–1522, Dec. 15, 2021.

[14] S.-Y. Kim, T. Suzuki, J.-I. Kani, and T. Yoshida, "Carrier phase estimation softwarized on GPU using decision-aided phase unwrapping for flexible optical coherent access systems," *J. Lightw. Technol.*, vol. 39, no. 6, pp. 1706–1714, Mar. 15, 2021.

[15] T. Suzuki, S.-Y. Kim, J.-I. Kani, and T. Yoshida, "Real-time polarization demultiplexing by multi-thread constant modulus algorithm for fully softwarized access networks," *J. Lightw. Technol.*, vol. 41, no. 5, pp. 1346–1356, Mar. 1, 2023.

[16] A. J. Viterbi and A. M. Viterbi, "Nonlinear estimation of PSK-modulated carrier phase with application to burst digital transmission," *IEEE Trans. Inf. Theory*, vol. IT-29, no. 4, pp. 543–551, Jul. 1983.

[17] T. Suzuki, S.-Y. Kim, J.-I. Kani, and T. Yoshida, "Real-time software implementation of coherent receiver DSP adopting multiplication-based parallel frequency offset compensation for fully virtualized access networks," in *Proc. Opt. Fiber Commun. Conf. Exhib. (OFC)*, Mar. 2023, pp. 1–3.

[18] L. Li, Z. Tao, S. Oda, T. Hoshida, and J. C. Rasmussen, "Wide-range, accurate and simple digital frequency offset compensator for optical coherent receivers," in *Proc. Conf. Opt. Fiber Commun./Nat. Fiber Optic Eng. Conf.*, Feb. 2008, pp. 1–3.

[19] A. Kelkar and C. Dick, "NVIDIA aerial GPU hosted AI-on-5G," in *Proc. IEEE 4th 5G World Forum (5GWF)*, Oct. 2021, pp. 64–69.

[20] *IEEE Standard for Information Technology—Telecommunications and Information Exchange Between Systems—Local and Metropolitan Area Networks—Specific Requirements—Part 3: Carrier Sense Multiple Access With Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications*, IEEE Standard 802.3av-2009.

TAKAHIRO SUZUKI (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in engineering from Waseda University, Tokyo, Japan, in 2012, 2014, and 2017, respectively. In 2014, he joined NTT Access Network Service Systems Laboratories, NTT Corporation, Kanagawa, Japan. From 2019 to 2020, he was involved in OSS development for SDN Enabled Broadband Access (SEBA), Open Networking Foundation (ONF), Menlo Park, CA, USA. His research interests include SDN/NFV for optical access networks and real-time systems using accelerators.

SANG-YUEP KIM received the Ph.D. degree in electronics engineering from Kwangwoon University, Seoul, South Korea, in 2004. From 2004 to 2007, he was with the University of Tokyo, Japan, under a Postdoctoral Foreign Researcher Fellowship. In 2008, he joined NTT Access Network Service Systems Laboratories, NTT Corporation, Chiba, Japan. He is currently researching DSP technologies and software-defined access architectures for future optical access systems.

JUN-ICHI KANI (Senior Member, IEEE) received the B.E., M.E., and Ph.D. degrees from Waseda University, Tokyo, Japan, in 1994, 1996, and 2005, respectively, all in applied physics. In 1996, he joined NTT Optical Network Systems Laboratories, where he engaged in researching optical multiplexing and transmission technologies. Since 2003, he has been with NTT Access Network Service Systems Laboratories, where he is engaged in the research and development of optical communication systems for metro and access applications, and currently heads the Access Systems Technology Group. He has been participating in ITU-T and the Full Service Access Network initiative (FSAN), since 2003.

TOMOAKI YOSHIDA (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in communication engineering from Osaka University, Osaka, Japan, in 1996, 1998, and 2007, respectively. In 1998, he joined the NTT Multimedia Systems Development Center, Japan, where he engaged in the development of ATM optical access systems. In 1999, he moved to NTT Access Network Service Systems Laboratories, Japan. Since 2000, he has been engaged in research on next-generation optical access networks and systems. From 2013 to 2015, he engaged in the research of WDM/TDM-PON and involved on the standardization of optical access systems, such as NG-PON2. Since 2020, he has been a Visiting Professor with the Graduate School of Information Science and Technology, Hokkaido University. He is currently the Project Manager of NTT Access Network Service Systems Laboratories. He is a member of IEICE, Japan.