CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

FPGA-based Wordlength Evaluation and Optimization for ASIC Implementation

Master's thesis in Embedded Electronic System Design

JINSHENG BIAN & CHENHAO YANG

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Master's thesis 2024

FPGA-based Wordlength Evaluation and Optimization for ASIC Implementation

JINSHENG BIAN & CHENHAO YANG

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2024

FPGA-based Wordlength Evaluation and Optimization for ASIC Implementation
JINSHENG BIAN & CHENHAO YANG

Supervisor: Per Larsson-Edefors, Department of Computer Science and Engineering
Co-Supervisor: Erik Börjeson, Department of Computer Science and Engineering
Examiner: Lena Peterson, Department of Computer Science and Engineering

Master's Thesis 2024
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2024

FPGA-based Wordlength Evaluation and Optimization for ASIC Implementation
JINSHENG BIAN & CHENHAO YANG
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

DSP designs are widely implemented by fixed-point representations for the consideration of hardware resource utilization. However, determining appropriate fractional wordlengths for signals is difficult as it requires balancing accuracy and hardware cost. Traditionally, wordlength optimization (WLO) problems are solved by simulations to estimate signal accuracy, but this process can be slow for complex DSP designs. To achieve fast convergence speed in more intricate DSP circuits, we propose a modified variant of the tree-structured Parzen estimator (TPE) algorithm, which is commonly used in hyperparameter optimization. By streaming data in hardware, FPGA emulation significantly outpaces software simulation. Therefore, we introduce an FPGA-accelerated WLO system utilizing FPGA emulation. We implemented our system on three DSP designs: two finite impulse response (FIR) filters and one phase recovery design. The results show a dramatic reduction in evaluation time for signal accuracy, achieving speedup factors of 504 for the 29^{th}-order FIR filter, 342 for the 14^{th}-order FIR filter, and 487 for the phase recovery design.

Keywords: Wordlength optimization, FPGA, DSP, TPE, SW/HW co-simulation

Acknowledgements

We would like to thank our supervisor, Per Larsson-Edefors, and co-supervisor, Erik Börjeson, for providing the topic and administrative and technical guidance and help during the project. We also appreciate the advice from our examiner, Lena Peterson, and the server permission from Lars Svensson.

Jinsheng Bian & Chenhao Yang, Gothenburg, July 2024

Contents

1

Introduction

The rapid improvement in the transistor density on the chip enables more functionalities to be implemented on hardware such as application-specific integrated circuits (ASICs) and field programmable gate arrays (FPGAs). Digital signal processing (DSP) applications on such platforms usually involve casting floating-point operations into fixed-point operations for the consideration of hardware limits such as area, power, and resources. Unlike floating-point operations where the operands are first decoded and then fed into different arithmetic calculations, fixed-point operations are consistent with general binary integer operations. The fixed point numbers have integer and fractional parts, whose lengths can be adjusted according to the design specification. However, substituting floating-point operations with fixed-point operations brings a reduction in the quality of the result due to less accuracy. Therefore, the optimization problem is raised to find the best wordlength configurations so that the design can meet the accuracy specification at the minimum cost. Similar to other system optimization problems, the wordlength optimization (WLO) can be described as:

$$C_{min}(W) \; subject \; \rightarrow \; \lambda(W) \geq \lambda_{min} \tag{1.1}$$

which means the cost is expected to be minimal with the constraint of given lower bound accuracy, where W, C, and λ represent the configuration, the cost, and the accuracy, respectively.

However, estimating the cost and the accuracy are difficult for complicated designs. Instead, simulation-based approaches in which massive simulations and synthesis runs are used to achieve the optimal wordlength configurations to balance the cost and the accuracy [1]. There have been significant researches on the WLO problems [2, 3, 4], and they are all utilizing the simulation-based approach with different optimization algorithms. The simulation and the synthesis can be both time-consuming for complicated designs. To speed up the optimization process, the concept of FPGA acceleration was proposed to accelerate the simulation [5]. In this project, we aim to implement a framework with an improved heuristic algorithm for faster convergence time, to carry out WLO on DSP designs by utilizing FPGA acceleration.

1.1 Related Work

There are many WLO methods proposed in the literature, which can be categorized into two types: analytical approaches and simulation-based approaches [1].

To obtain the optimal wordlength configurations, analytical approaches solve a cost function that empirically or theoretically models the system based on some metrics (e.g. accuracy, area, and power), whereas the simulation-based approaches conduct iterative search with simulations. However, it is hard to find an accuracy function for a complex design consisting of nonlinear blocks, which makes obtaining the optimal solution using an analytical approach unfeasible. Meanwhile, the simulation-based approach selects one local optimum from the solution space, which expands exponentially as the number of parameters grows and usually leads to a long simulation time. Choi and Burleson have initially addressed this problem in [1] and proposed a search-based framework for simulation-based approaches, based on which various algorithms have emerged. For instance, Cantin et al. proposed the greedy deterministic algorithms: Min+1 and Max-1 in [2] to obtain the best wordlength configuration by varying the wordlength in the step of 1 bit. Nguyen et al. combined the two methods above and Tabu search in the proposed two Greedy Randomized Adaptive Search Procedure (GRASP) based algorithms [3], which allow searching in both descent and ascent directions. Some other classic algorithms are modified and utilized in WLO problems, such as Adaptive Simulated Annealing (ASA) [4].

Meanwhile, Bayesian optimization (BO) has emerged as a powerful tool for solving the problem (1.1), and is a global optimization strategy constructed with the guidance of Bayes' Rule, where the estimated model is evaluated and updated with iterations, and the optimum is achieved during the iterative process. BO was first applied to optimize the wordlength in [6]. Based on (1.1), Ha et al. have proposed a resource-constrained BO scheme in [7] to maximize computing accuracy with the restriction of resources, which is proven to significantly outperform the Tabu search algorithm [3].

Simulation-based approaches generally suffer from long simulation time, and FPGA-based acceleration for emulating the DSP design was proposed by Hormigo and Caffarena in [5] for WLO purposes. In [5] the DSP design was implemented on an FPGA and its wordlengths were controlled by the precision limiter. Therefore, the wordlengths could be varied during runtime. Additionally, plenty of FPGA-emulated DSPs in the other studies can also be used for WLO. For instance, Börjeson and Larsson-Edefors proposed an FPGA-based fiber emulator to emulate the optical impairments such as polarization mode dispersion in [8], which enables users to run long-term emulation of their DSP designs.

1.2 Purpose and Goal

The purpose of the project is to develop a hybrid WLO system on FPGA and PC to optimize the wordlengths for ASIC implementation of DSP designs. The hybrid system will consist of hardware and software, where the hardware is used to carry out the emulation of DSP designs and preprocess the results, the software is used to implement the optimization algorithm and configure the hardware. As a result, the overall system will be able to find the optimal wordlength configurations for the predefined loss function. To be more specific, the project has several goals:

- Develop a program that carries out the optimization algorithm and controls

the hardware and carries out the synthesis to extract power and area results for the ASIC implementation.

- If possible, improve the optimization algorithm for the search efficiency and the quality of the solution.
- Develop a hardware platform where the emulation result of the DSP design is preprocessed on FPGA and then transmitted into software for the optimization algorithm.
- Compare and contrast the emulation-based method with the simulation-based method.

1.3 Considerations in Ethical and Ecological Aspects

Our project targets ASIC wordlength optimization. In the aspect of ecology, optimized wordlength enables circuits to consume less power and chip area, which contributes to environmental protection. In ethical aspects, our project involves utilizing others' IPs such as algorithms and tools, we have to ensure that all processes respect others' IP rights.

1.4 Thesis Outline

This report is organized according to the following structure: Chapter 2 introduces the related background information including Bayesian optimization. Chapter 3 presents the methods used in the design flow. Chapter 4 illustrates how the hybrid system is constructed in detail. Chapter 5 shows the results obtained in our work and carries out comparisons. In Chapter 6, we show the discussions on the results and future work. Last, we conclude the project in Chapter 7.

2

Technical Background

This chapter aims to introduce the background knowledge for the project, which mainly involves digital signal processing systems, hyperparameter optimization, especially Bayesian Optimization, and general techniques for wordlength optimization.

2.1 Digital Signal Processing Systems

A digital signal processing (DSP) system is generally used to perform a wide range of mathematical operations to process signals. This is usually designed with the help of software and can be implemented on both software and hardware platforms. The software implementation has good flexibility, while the high power efficiency and throughput brought by the hardware implementation have attracted extensive attention. There are two common platforms where we can turn a DSP system into hardware: application-specific integrated circuit (ASIC) and field programmable gate array (FPGA). As DSP systems become more and more complex in modern industrial settings, ASIC has been widely used to implement digital hardware systems because of its low power consumption and high speed. However, high non-recurring engineering (NRE) cost which comes from masks and wafer changes makes it not suitable to be used in small-scale manufactures. FPGA is a special type of ASIC that consists of many components such as lookup tables, flip-flops, and DSP slices, which make it reconfigurable. It can be configured to perform any digital functions, which greatly reduces the NRE cost although usually brings a higher power consumption. So it is a good alternative to ASIC, especially in cases where only a small number of implementations are required.

Logic simulation is a way to verify the functionality of a digital design including DSP systems, which generally involves feeding some test vectors to the design and checking the correctness of the output sequence. For simple DSP systems, the accuracy of output signals can be quickly evaluated by simulations. However, for complex digital systems, the simulation tends to be slow, which makes the simulation-based WLO time-consuming.

2.2 Wordlength Optimization

Different from the generalized computing with floating-point numbers in software, hardware implementations of DSP algorithms often utilize fixed-point arithmetic

to save on-chip resources, which brings the need for WLO. A fixed-point number comprises an integer part (representing dynamic range) and a fractional part (representing accuracy). While optimizing the dynamic range to avoid overflow and save on-chip resource simply involves static analysis or simulation [3], optimizing accuracy poses a significant challenge, that is, the loss function is a black box with an exponentially growing search space as dimension increases. Here, the dimension refers to the number of signals whose wordlengths need to be optimized. Heuristic search can take guided routines based on the evaluation of current samples. As a consequence, in terms of efficiency, heuristic search outperforms exhaustive search for the search space in high dimensions [9]. One key component in the heuristic search is the loss function, based on which the guiding policy is carried out. The loss function is affected by the metrics that are chosen to evaluate the system. Common metrics like signal-to-noise ratio (SNR), power consumption, area usage, bit error rate (BER), mean square error (MSE), etc., provide evaluations from various perspectives. The loss function may comprise multiple metrics.

2.3 Bayesian Optimization

The machine learning-based optimization method, called Bayesian optimization (BO) [10], is widely used in the problem where the loss function is a black box. Simply, BO method can estimate the loss function based on the given data and find the optimum value by updating the predicted loss value iteratively. The optimization process mainly consists of two parts: a surrogate model to estimate the loss function, and an acquisition function to select the next query point we want to observe. The surrogate model can be either parametric or non-parametric [10]. The former is applied when the problem can be modeled by a certain probability distribution, which make it easier to calculate compared to the latter where a stochastic method such as Gaussian process or random forest is used to represent the predicted value and its uncertainty. The acquisition function is used to select the next query point based on the given data. The overall workflow of BO can be described by Figure 2.1. It should be noted that the exit criterion used in the workflow is the quality of the solution (better quality means less loss value), which can also be substituted by other conditions like the number of iterations.

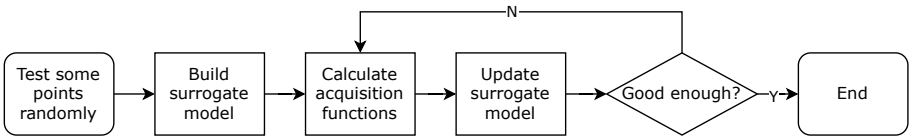


Figure 2.1: The main workflow of BO.

2.3.1 Bayes' Rule

The update of the model in BO follows Bayes' Rule, described by the equation below:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)} \tag{2.1}$$

where y is the loss function to be optimized and x is the input data of the function, $p(x)$ is the marginal likelihood (or evidence) to normalize the probability distribution, $p(y)$ is the prior estimation of the loss function that shows the predicted value and uncertainty, $p(x|y)$ is the likelihood, and $p(y|x)$ is the posterior estimation of the loss function. Usually, to simplify the calculation, the conjugate prior which will lead to the posterior in the same distribution family is always selected[10].

2.3.2 Surrogate Model

As BO focuses on solving optimization problems where obtaining the exact numerical expression of the loss function is not feasible, it employs a surrogate model to depict the behavior of the loss function. This surrogate model is continually updated with newly observed data. Given the inherent uncertainty of the loss function, non-parametric models are commonly employed in general optimization scenarios, like Gaussian process regression in the hyperparameter tuning problems.

2.3.3 Acquisition Function

The acquisition function plays a vital role in determining the next query point before model update. Various acquisition functions prioritize distinct search objectives, such as exploitation and exploration. The acquisition function used in this thesis is Expected Improvement (EI) [11], described by the equation below:

$$E[I(x)] \equiv E[max(f_{min} - Y, 0)] \tag{2.2}$$

where $I(x)$ is the improvement at point x, f_{min} is the minimum value from the observed data, and Y is the newly observed data. EI can serve as a policy during the iteration to identify the subsequent query point that aligns with its maximum.

2.4 Tree-structured Parzen Estimator

As described in section 2.3, surrogate models with stochastic steps are used to model the overall loss function. Take the Bayes' Rule (Equation 2.1) for instance, in which the conventional surrogate models construct the estimation of the posterior distribution $p(y|x)$. Nevertheless, computing the posterior in high dimensions requires a significant amount of calculations. Tree-structured Parzen Estimator (TPE) proposed in [12], is a new surrogate model to estimate the likelihood $p(x|y)$ and the prior $p(y)$. The likelihood in TPE is defined as:

$$p(x|y) = \begin{cases} l(x) & y < y^* \\ g(x) & y \geq y^* \end{cases} \tag{2.3}$$

where the probability distribution of observed data is divided into two sub distributions, $l(x)$ where the loss value is less than y^* and $g(x)$ where the loss value is greater than y^*, and y^* is the dividing criteria which is defined by the quantile γ where the cumulative probability $p(y < y^*) = \gamma$.

Applying EI to TPE, the acquisition function obtained is [12]:

$$\begin{aligned} E(x) &= \int_{-\infty}^{y^*} (y^* - y) p(y|x) dy \\ &= \frac{\gamma y^* l(x) - l(x) \int_{-\infty}^{y^*} y p(y) dy}{\gamma l(x) + (1-\gamma) g(x)} \propto \ (\gamma + \frac{g(x)}{l(x)}(1-\gamma))^{-1} \end{aligned} \tag{2.4}$$

Therefore, the next query point can be obtained by finding the maximum value of $\frac{l(x)}{g(x)}$. Moreover, the massive calculations on the posterior are avoided because only the likelihood is needed for the selection, which reduces the intervals during iterations, especially in high dimensions.

2.4.1 Kernel Density Estimation

Except the acquisition function, the methods to construct the probability function also affect the overall performance significantly. An intuitive way to construct the model of $l(x)$ and $g(x)$ is using the Gaussian mixture model [13], described as:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{2.5}$$

where π_k is the mixing coefficient, satisfying $\sum_{k=1}^{K} \pi_k = 1$, and $\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is the component Gaussian distribution with its mean $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$.

However, refining the Gaussian mixture model with Expectation Maximization [14] is also an iterative process that would increase the interval during iterations. Instead, Kernel Density Estimation (KDE) [15] is used to construct $l(x)$ and $g(x)$ by summing up the identical Gaussian kernel of all samples in $l(x)$ and $g(x)$ separately, described as:

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right) \tag{2.6}$$

where h is the window width, and $K\left(\frac{x-X_i}{h}\right)$ is the kernel function at X_i.

There are two categories of Gaussian kernels: the univariate and the multivariate. Despite requiring more computations, the multivariate kernel is selected for the project due to its ability to yield improved modeling outcomes, as demonstrated in [16].

3

Methods

The detailed workflow of the project is illustrated in this chapter.

3.1 Project workflow

The workflow of this project mainly consists of two parts, the algorithm and the system. For the algorithm, the work is organized as:

- Reference WLO algorithm selection
- Improvement for the reference algorithm
- Algorithm comparison

After the algorithm part, the improved algorithm and the reference algorithm will be implemented in the system, which is organized as:

- Hybrid WLO system framework construction
- Hybrid WLO System implementation
- WLO system comparison

3.2 Algorithm

First, an optimization algorithm will be selected and later implemented as a reference. A hyperparameter optimization algorithm is theoretically designed under the assumption of ideal conditions in every aspect. However, implementing such a complex algorithm in software usually requires storing large amount of values. It is usually infeasible because of the computer performance limitations, which creates some drawbacks to the quality of the result. So in the following section, we will identify the drawbacks of the reference algorithm implementation and improve the algorithm's quality at a low cost by applying some machine-learning methods. Then the improved algorithm will be compared with the reference one. The algorithms on some typical mathematical objective functions will be compared in terms of the quality of solutions with the same number of iterations.

3.3 Hybrid System

While testing the algorithm, the framework of the hybrid WLO system will be constructed and compared to the reference WLO system, which is simulation-based.

The hybrid system framework is expected to be used for various DSP designs. In the simulation-based WLO optimization, a simulation software such as Modelsim can be used to evaluate the accuracy of the output and a synthesis tool such as Genus with a technology library can be used to assess the cost (e.g. power or area). For the hybrid WLO system, the simulation part will be replaced by the FPGA emulation. That is, the hybrid WLO system will be implemented, containing the modified algorithm guiding the optimization process and an accuracy evaluation hardware module working on FPGA. The hardware part is expected to control the signal wordlengths of the DSP design in real time and return the accuracy evaluation result to the optimization algorithm. The electronic design automation (EDA) tool (i.e. Cadence Genus) will be used to generate the power or area reports for the cost evaluation. The ASAP7 [17] technology library will be used for area estimation in the project.

Finally, in terms of the optimization time, convergence speed, and solution quality, the hybrid system will be compared to the reference system with some different DSP designs, where the reference system is the system utilizing the same framework as the hybrid system but including the reference algorithm.

4

Design

Following the introduction of the reference system design, this chapter presents the detailed designs of each level in the hybrid system.

4.1 Reference System Design

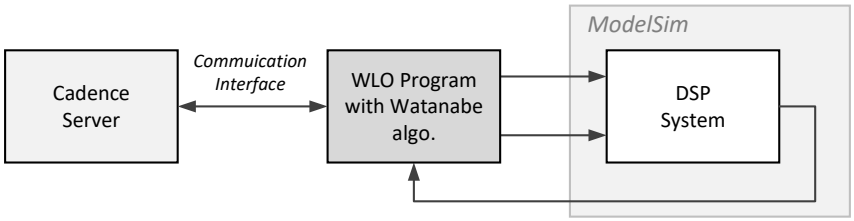


Figure 4.1: Structure of the simulation-based WLO system.

The reference WLO system (i.e. the simulation-based WLO system) for the DSP design is shown in Figure 4.1. The reference WLO program in the structure is the TPE developed by Watanabe [16] (noted as Watanabe's TPE), based on which a framework for simulation-based WLO is constructed. The reference WLO program obtains data from both the synthesis tool and the hardware simulated by Modelsim. In each iteration, the WLO program sends the input data with changed wordlength to the DSP system, and the output is read by the program, based on which the WLO program calculates the mean squared error (MSE). The EDA tool (i.e. Cadence Genus) synthesizes the DSP design and generates power or area results for the given wordlength configuration. Then the WLO program calculates the loss function based on the collected data, which helps construct the surrogate model and compute the acquisition function. The time spent on the optimization process and the value of the loss function for the optimized wordlength configuration will be recorded for comparison.

4.2 Hybrid System Design

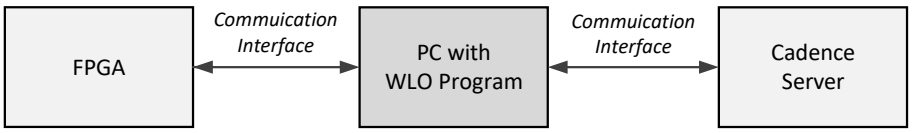


Figure 4.2: Overall design for our WLO system.

Figure 4.2 illustrates the hybrid WLO system, which mainly involves three parts:

- An FPGA where the target DSP system is emulated and the accuracy of the output data is evaluated.
- A PC where the WLO program is running.
- A server with Cadence running on it to generate the power reports and area reports.

The PC with the WLO program transmits wordlength configuration data to the FPGA and receives the preprocessed output via UART, and it also collects area reports from the synthesis tool. Based on the results, the program calculates the loss function and determines the next query point. The optimal wordlength configuration with the lowest loss is gradually approached after running a series of iterations.

4.3 Software Design

The software part in the hybrid system mainly contains three parts: the algorithm component to carry out the optimization, the synthesis component collecting the synthesis result, and the accuracy component to receive accuracy estimation result from hardware, as shown in Figure 4.3. Each part of the software will be introduced in detail in the following.

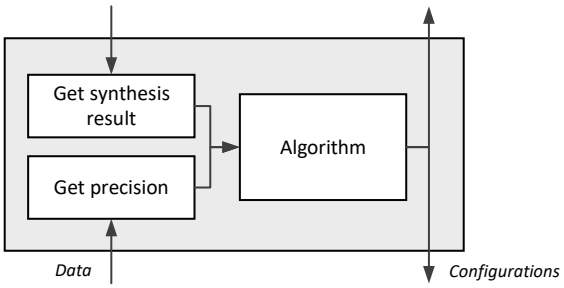


Figure 4.3: The structure of the software part

4.3.1 Tree-structured Parzen Estimator with batch mode

The algorithm which controls the optimization is described by Algorithm 1. In Algorithm 1, steps 1 to 6 are the startup process, and steps 7 to 20 are iterative optimizations which can be divided into two parts, the exploration and the exploitation as described in Algorithm 2 and 3.

The main feature that differs from other TPEs is the batch mode, by which the algorithm is able to evaluate several points in each iteration. For exploration, a batch of points are selected near the mean points of the good observations. But for exploitation, a batch of points are generated by equally dividing the distance between P_{max} and P_{best}, as described in Algorithm 3.

Algorithm 1 Tree-structured Parzen estimator (TPE)

Require: N_{init} (The number of initial evaluations), N_{total} (The number of evaluations other than N_{init}), S (The search space), F (The loss function), B (The batch size), λ (The quantile to split the observations), N_{sample} (The number of samples when evaluating the acquisition function).

$\mathcal{D} \leftarrow \emptyset$
for $n = 1, 2, \ldots, N_{\text{init}}/B$ **do**
    Randomly pick x_n (containing B configurations) within S
    $y_n := F(x_n)$ ▷ Evaluate the loss function
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(x_n, y_n)\}$ ▷ Initialization
end for
for $n = 1, 2, \ldots, N_{\text{total}}/B$ **do**
    Sort D according to the values of y_n
    Split $\mathcal{D}$ into $\mathcal{D}^{(l)}$ and $\mathcal{D}^{(g)}$ by a quantile λ
    Construct the acquisition function
    Sample the acquisition function near the mean point of $\mathcal{D}^{(l)}$ to N_{sample} points and get the configuration x_{max} resulting in maximum acquisition function value
    Applying SGD to further improve the x_{max} and collect the intermediate points $x_{intermediate}$
    if T is too dense at x_{max} **then**
        Exploration
    else
        Exploitation
    end if
    $y_{N+B} := f(x_{N+B})$ ▷ Evaluate the loss function
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(x_{N+B}, y_{N+B})\}$
end for

Algorithm 2 Exploration

Require: w (The weight within (0,1), B (Batch size))

$x_{tmp} = x_{mean}$ ▷ Get the mean points in $D^{(l)}$
$X_{N+B} = NULL$ ▷ Initialize the batch configurations to null
$Refresh\ limit = 10 + B$
while X_{N+B} is not full **do**
    if $Refresh\ limit == 0$ **then**
        if x_{tmp} **then** is observed
            Generate a random point x_{rand} within the search space
            $x_{tmp} = (1 - w) * x_{mean} + w * x_{rand}$
        else
            Add x_{tmp} into X_{N+B}
        end if
    else
        if x_{tmp} **then** is observed
            Add random move to x_{tmp}
            $Refresh\ limit -= 1$
        else
            Add x_{tmp} into X_{N+B}
            $Refresh\ limit += 1$
        end if
    end if
end while
Return X_{N+B}

Algorithm 3 Exploitation

Require: B (The batch size), $x_{intermediate}$ (The intermediate points during SGD), x_{max} (The maximum point obtained from SGD), x_{best} (The best point in $D^{(l)}$)

for i in $x_{intermediate}$ **do**
    if x_{max} is observed **then**
        $x_{max} = i$
    else
        Break
    end if
    $X_{N+B} = (x_{max} - x_{best})/B * n,\ n = 1, 2, \ldots, B$
end for
Return X_{N+B}

Gaussian mixture model

The Gaussian mixture model (GMM) of $D^{(l)}$ is constructed by the unevenly weighted sum of multivariate Gaussian distributions, corresponding to step 10 in Algorithm 1. Step 10 contains two parts: build the probability function of $D^{(l)}$ and $D^{(g)}$, and build

the acquisition function. Here corresponds to the former. The weights are:

$$w = \frac{2(N-n)}{N(N+1)}, \quad n = 0, 1, \ldots, N-1 \tag{4.1}$$

where N is the number of observations in $D^{(l)}$, and the observations in $D^{(l)}$ are ordered from best to worst and indexed from 0 to $N-1$. Therefore, the better observations have larger weights. And the GMM of $D^{(g)}$ is constructed by the evenly weighted sum of the multivariate Gaussian distributions, that is, the sum is divided by the number of observations in $D^{(g)}$.

The expression of multivariate Gaussian distribution is:

$$p(x \mid \mu, \Sigma) = \frac{1}{(2\pi)^{k/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right) \tag{4.2}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the mean value and the co-variance matrix. To reduce the computation complexity, the covariance matrix is a diagonal matrix with equal diagonal items. Therefore, the calculation of the probability at x can be described as:

$$p(x|\mu, \sigma) = \sum_{n=0}^{N-1} \frac{1}{\sqrt{2\pi\sigma^2}} \exp(\frac{d_n^2}{2\sigma^2}), \; d_n = x - \mu[n] \tag{4.3}$$

where μ is the point set of $D^{(l)}$, $\mu[n]$ is the $(n+1)^{th}$ point in $D^{(l)}$, and σ is the value in the diagonal of the covariance matrix in Equation 4.2. In this project, the σ is set to one-eighth of the maximum range in the search space.

Acquisition function

As illustrated by Equation 2.4, the next query point is where the maximum ratio of the observations $g(x)$ and $l(x)$ is. However, with the dimension increasing, the values in the GMMs within the search space increase exponentially. Therefore, it is not feasible to calculate all values and get the maximum. Instead, other methods such as sampling [16] are used to approximate the maximum. In this project, to further improve the sampling, the stochastic gradient descent-ascent (SGDA) [18] is used, corresponding to step 12 in Algorithm 1. Since only the maximum is investigated, only the ascent feature is used here, described by the equation:

$$x = x + \eta\nabla_x f(x) \tag{4.4}$$

where x is the configuration corresponding to the lower case x in the algorithms, η is the learning rate, $\nabla_x f(x)$ is the gradient at x.

The learning rate in SGDA is dependent on the loss function. In WLO problems, the configurations x are integers and only vary within a small range. Thus, to limit the iteration speed, the update in the SGDA is modified to:

$$\hat{u} = \text{SIGN}(u) \times \log(|u| + 1), \; u = \eta\nabla_x f(x) \tag{4.5}$$

It should also be noted that the start point of SGAD is the point near the mean point of $D^{(l)}$, which is obtained by:

$$P_{start} = w \times P_{random} + (1-w) \times P_{mean} \tag{4.6}$$

where w is the weight value between 0 and 1, P_{random} is a random point in the search space, and P_{mean} is the mean point of $D^{(l)}$.

Division of Exploration and Exploitation

In Algorithm 1, step 13 is the conditional argument to determine if the next step is exploitation or exploration. In this project, the density of one point is calculated by the probability of the GMM of $D^{(l)}$. The critical value is calculated by Equation 4.7, which is generated by approximating empirical data. If the density is larger than this value, the next point will be targeted for exploration.

$$B = \frac{1}{\sqrt{2\pi\sigma^2}} \times \exp(-0.09(N-0.5)^{1.4}) \times \exp(-(1/(10N_{eval})(l-0.99))^{0.3}) \quad (4.7)$$

where N is the dimensions, σ is the standard deviation in Equation 4.2, N_{eval} is the total number of evaluations, and l is the number of items in $D^{(l)}$.

4.4 Loss function

Since several metrics such as accuracy and area need to be optimized, the loss function reflects how well the metrics are optimized. This project uses the product of the absolute values as a loss function to combine those metrics, where better accuracy and area occupation always lead to a lower multiplication result. Since synthesis may take a long time, a range for the accuracy metric is set for the avoidance of the synthesis on unreasonable wordlength configurations. The loss function is described as:

$$loss = \begin{cases} |accuracy - target| \times area & lower\ bound \leq accuracy \leq upper\ bound \\ |accuracy - target| \times C & \text{otherwise} \end{cases} \quad (4.8)$$

where *accuracy* is the accuracy metric, *target* is the optimization target for the accuracy and in the range, *area* is the area from synthesis, C is a large constant (larger than the area when the accuracy is in the range), *lowerbound* and *upper bound* define the range of the accuracy

4.5 Hardware Design

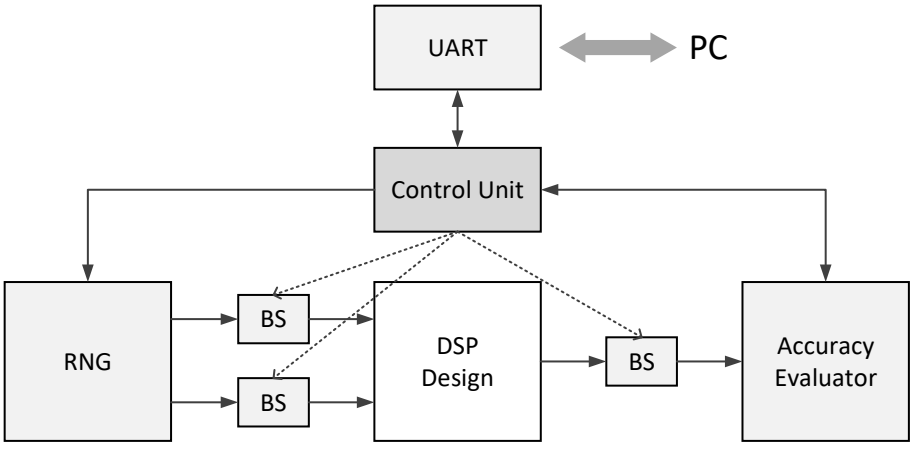


Figure 4.4: General hardware structure for FPGA-based hybrid WLO system.

The block diagram for the hardware system running on FPGA is shown in Figure 4.4, which typically consists of the following modules:

- UART: the interface communicating with the PC.
- Control Unit: a module that receives wordlength configuration for the DSP system from the PC, and sends the preprocessed output data (such as the MSE of the output sequence) to the PC.
- Random Number Generator (RNG): a module that stores the input sequences for the DSP system.
- Bit Switch (BS): modules that control the wordlength of the data path by masking.
- DSP System: the target digital design where we want to optimize the wordlengths.
- Accuracy Evaluator: a module that collects output data from the DSP system and calculates the accuracy (e.g. MSE or BER) of the output.

Here is the general workflow of the hardware part: At first, the PC sends wordlength configurations to the FPGA. The Control Unit receives the values and distributes them to the Bit Switches. Then a start command is sent to the Control Unit, and the Accuracy Evaluator starts to collect the output values of the DSP system with Bit Switch masking. It will compare the data sequence with the reference sequence to evaluate the accuracy, which is measured by MSE. Finally, the evaluated accuracy values are sent to the PC.

4.5.1 Standard Mode

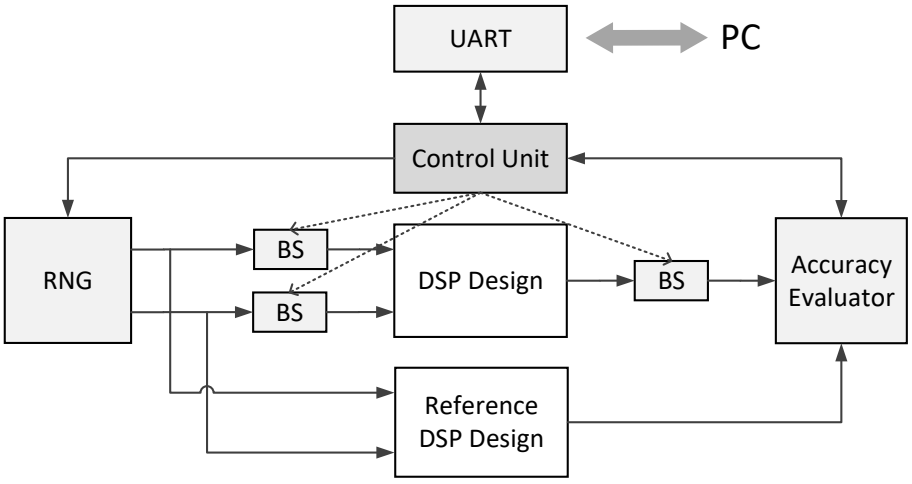


Figure 4.5: Typical hardware structure for FPGA-based hybrid WLO system.

For the Accuracy Evaluator module, it is possible to store the reference output sequence in the memory, but this method usually requires a lot of on-chip resources if the sequence is long. So we make two DSP designs running in parallel on FPGA (Figure 4.5), where one is used to obtain the reference sequence and another is used to generate the output with altered wordlengths. Then the accuracy Evaluator calculates the MSE based on the two sequences.

4.5.2 Batch Mode

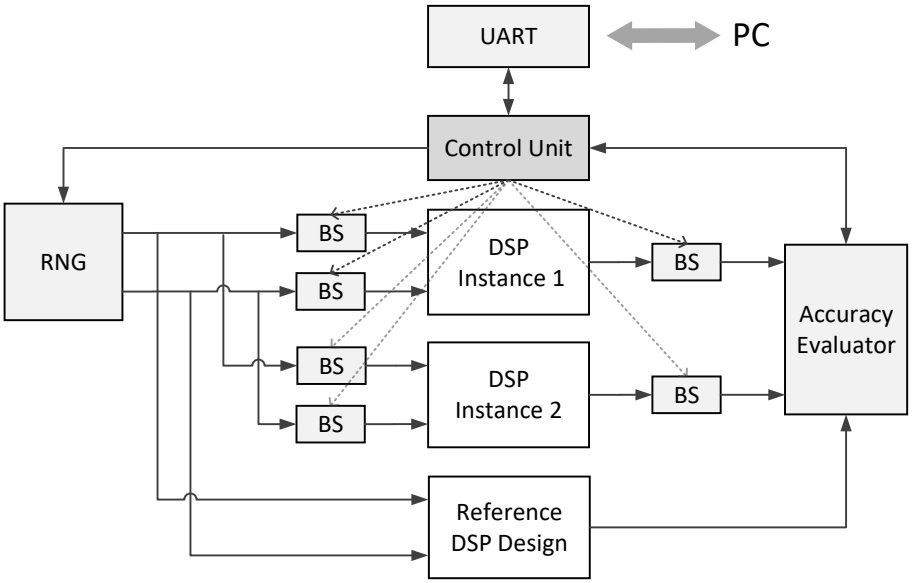


Figure 4.6: Batch mode (batch size = 2) hardware structure for FPGA-based hybrid WLO system.

To explore a further increase in the optimization speed, we propose the Batch Mode which can obtain multiple accuracy values. Figure 4.6 shows the batch mode block diagram with a batch size of 2. DSP 1 and DSP 2 receive different wordlength configurations, and the two output sequences are evaluated simultaneously.

5

Results

In this chapter, the results of the project will be presented in terms of the comparison of the algorithm and the comparison of the FPGA-emulation-based and simulation-based approaches.

5.1 Comparison of Algorithms

5.1.1 Benchmarks

In the comparison of algorithms, several benchmarks for optimization problems are selected to test the speed of convergence and the quality of the solution. We use the TPE developed by Watanabe in [16] and the Optuna developed by Akiba et al. in [19] are selected for references. The benchmarks for the comparison of the algorithms are shown in Table 5.1. The dimensions are set to $5D$, $10D$, $15D$ and $30D$. The Rastrigin function and Styblinski-Tang function are multimodal functions that contain multiple local minimums, and the Rosenbrock function and sphere function are monomodal functions that contain only one global minimum.

Table 5.1: Parameters of benchmarks.

Benchmark	Equation
Rastrigin	$\sum_{i=0}^{N-1} \left(\frac{x_i}{3}\right)^2 - 10\cos(2\pi x_i/3) + 10N$
Rosenbrock	$\sum_{i=0}^{N-2} 100\left(x_{i+1} - x_i\right)^2 + (1 - x_i)^2$
Sphere	$\sum_{i=0}^{N-1} x_i^2$
Styblinski-Tang	$\sum_{i=0}^{N-1} \frac{(x_i/3)^4 - 16(x_i/3)^2 + 5(x_i/3)}{2}$

Benchmark	Search Space	Optimum
Rastrigin	$-16 \leq x_i \leq 16$	0 when $x_i = 0$
Rosenbrock	$-16 \leq x_i \leq 16$	0 when $x_i = 1$
Sphere	$-16 \leq x_i \leq 16$	0 when $x_i = 0$
Styblinski-Tang	$-16 \leq x_i \leq 16$	$-39 \times N$ when $x_i = -9$

5.1.2 Comparison Results

As mentioned in the previous section, there are several parameters in the TPE-WLO. The main components of TPE-WLO are the construction of the Gaussian mixture

model (GMM) and the acquisition function. In Chapter 4, the standard deviation of GMMs is set to one-eighth of the maximum search range, and more fractions of the maximum search range are tested in this chapter. Another feature of of TPE-WLO is the stochastic gradient descent and ascent (SGDA), and its start point which is set by a point near the mean point of $D^{(l)}$. In addition, the learning rate of SGDA is scaled. To see the differences made by SGDA and its start point, several tests are carried out on the TPE-WLO with and without SGDA and random start point, and different learning rates. Finally, the different batch sizes of TPE-WLO are tested.

The configurations of reference algorithms are necessary. For Watanabe's TPE, the number of initial and total evaluations are set to 15 and 200, the minimum bandwidth factor is set to $1e{-}2$ (the default value), and the number of the acquisition function samples is set to 50. For Optuna, the sampler is set to the base sampler by default, and the number of total evaluations is set to 200. These settings are used in all tests.

To reduce the uncertainty of random numbers, the results are calculated by the average of the results of 20 runs. In summary, the tests are categorized by the parameters under investigation:

- Different standard deviations.
- With and without SGDA or random sampling.
- Different learning rates for SGDA.
- Different batch sizes.

Tests on different standard deviations

Figure 5.1 shows the iterations of the reference algorithms and TPE-WLO with different standard deviations. In these tests, the learning rate of SGDA in TPE-WLO is set to 10, the SGDA is enabled, random sampling is disabled, and the batch size is set to 1 (when the coefficient is set to 1/8, this setting is called the standard setting, corresponding to the blue line in the figure). From the quality of the solution (the obtained loss value in the end), the coefficients of 1/8 and 1/6 can provide a better quality of solution in most cases, especially in the multimodal functions (ie. the Rastrigin function and the Styblinski-Tang function). Therefore, the coefficient is set to 1/8 in the following tests. It should be noted that the data of reference algorithms is consistent with the data listed in [16].

Tests on SGDA and random sampling

SGDA is used to further improve the next query point, as described in Chapter 4. The start point of SGDA is selected by sampling randomly near the mean point of $D^{(l)}$ (called non-random sampling). To investigate the effects brought by SGDA and the non-random sampling, some tests are carried out on TPE-WLO with and without SGDA and non-random sampling. Figure 5.2 shows the iterations of the reference algorithms and TPE-WLO with and without SGDA and non-random sampling (the red curve is the standard setting). In these tests, the learning rate of SGDA in TPE-WLO is set to 10, the coefficient of the standard deviation is set to

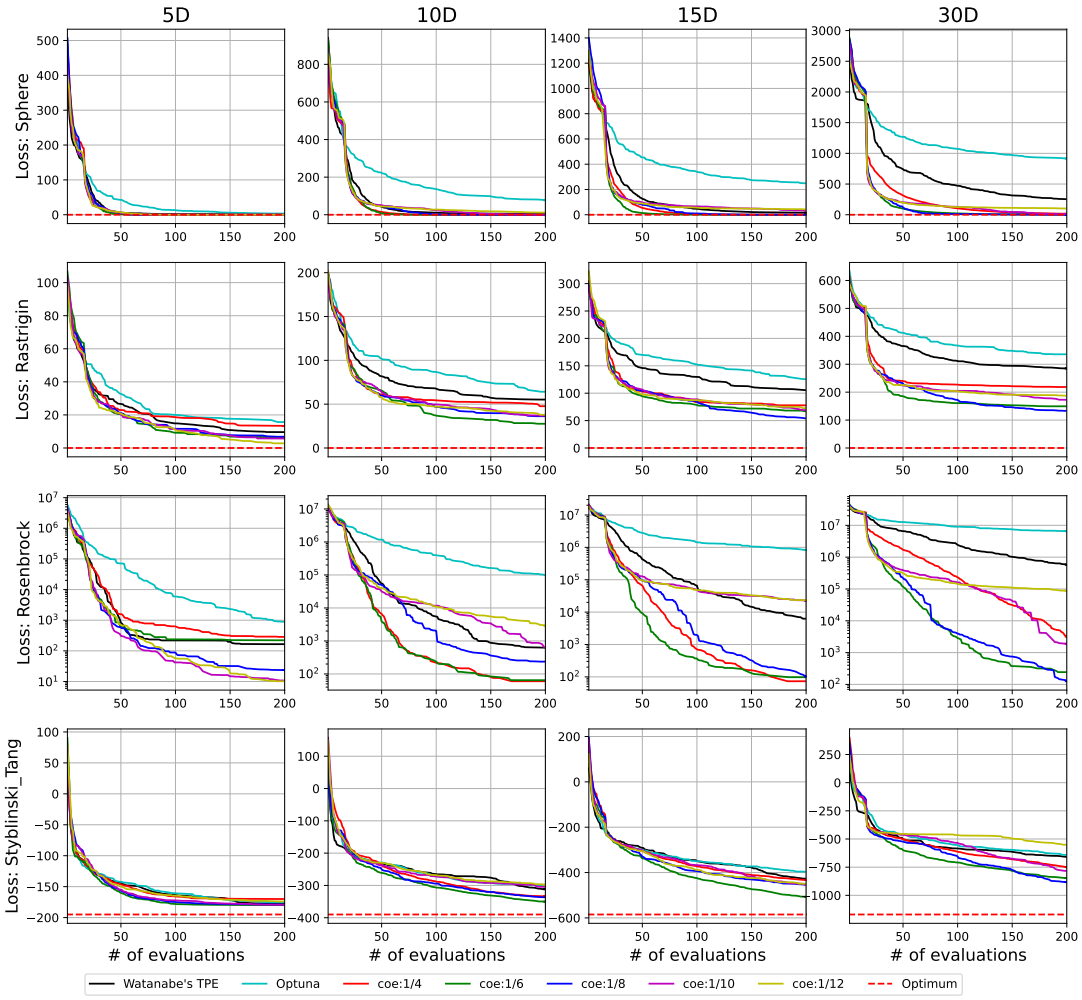


Figure 5.1: The tests on different fractions of the maximum search range. **coe** means the fraction of the maximum range, the product of the coe, and the maximum search range is the standard deviation of the GMM. The optimum of Rosenbrock is not shown due to the logarithmic coordinates. Each column has the same dimensions and each row has the same loss function (benchmark).

1/8, and the batch size is set to 1. It should be noted that in Figure 5.2, random means fully random sampling which is sampling randomly among the whole search space, opposite to the non-random sampling. From the result, non-random sampling improves the quality of solution significantly in Sphere function, Rastrigin function and Rosenbrock function, where SGDA has little impact on the quality of solution. However, in the Styblinski-Tang function, SGDA improves the quality of the solution further, so that WLO-TPE outperforms the reference algorithms in terms of convergence speed.

Tests on different learning rates

Although the learning rate is limited, it is still important for SGDA. In these parts, learning rates of $\{0.1, 1, 10, 20, 50\}$ are tested with SGDA, non-random sampling, and

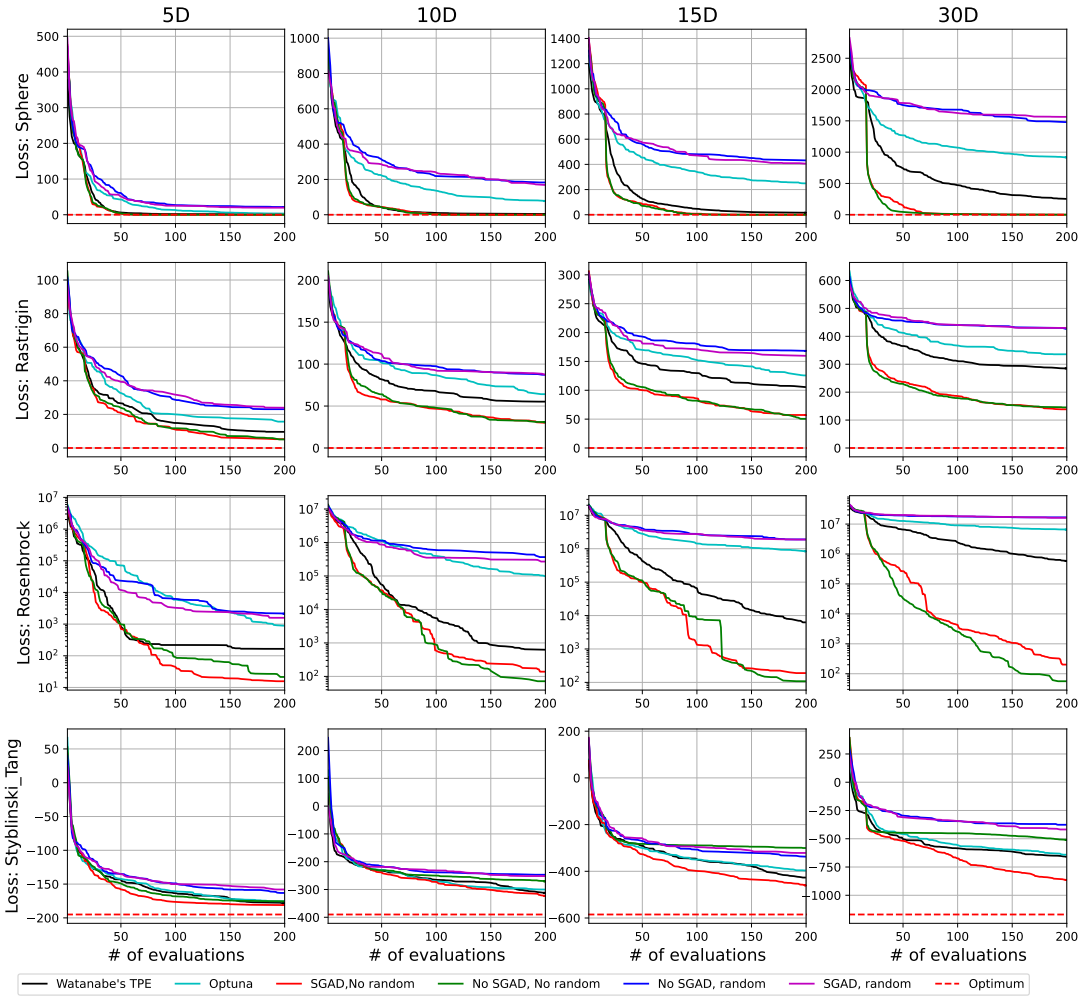


Figure 5.2: The tests on TPE-WLO with and without SGDA or random sampling. The optimum of Rosenbrock is not shown due to the logarithmic coordinates. Each column has the same dimensions and each row has the same loss function (benchmark).

the standard deviation fraction of 1/8. The iterations of WLO-TPE and reference algorithms are shown in Figure 5.3 (where the blue line is the standard setting). For the Sphere function and Rastrigin function, there are negligible differences among different learning rates. However, for the Rosenbrock function and Styblinski-Tang function, the quality of the solution is diverse with increasing dimensions. Among these results, the learning rate of 10 can provide a more stable performance than other values.

Tests on different batch sizes

From previous tests, the configurations for tests on different batch sizes are: the coefficient in the standard deviation is 1/8, the SGDA and non-random sampling are enabled, and the learning rate is set to 10. In these tests, the number of iterations is set to 400. Figure 5.4 presents the results on different batch sizes, and the red

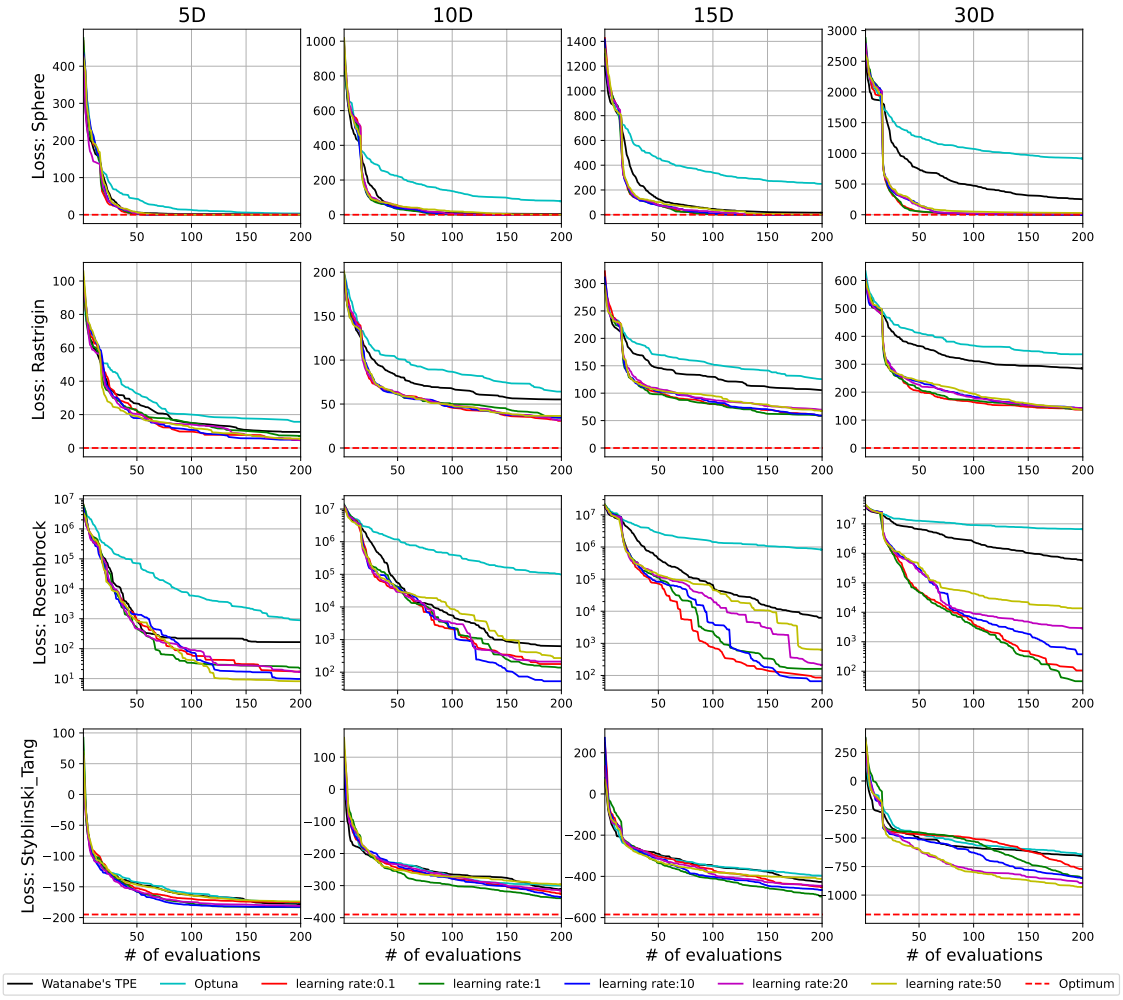


Figure 5.3: The tests on TPE-WLO with the learning rate of $\{0.1, 1, 10, 20, 50\}$. The optimum of Rosenbrock is not shown due to the logarithmic coordinates. Each column has the same dimensions and each row has the same loss function (benchmark).

curve is the standard setting in previous tests. For the Sphere function, Rastrigin function, and Rosenbrock function, all these batch sizes can provide a better quality of solution than the reference algorithms. However, for the Styblinski-Tang function, TPE-WLO can provide higher performance in dimensions 15D and 30D. The average time spent on different benchmarks in different dimensions is shown in Figure 5.5, time of different batch sizes in different dimensions remains almost the same, whereas the time of Watanabe's TPE and Optuna increases with the dimension increasing. And the time of evaluation for TPE-WLO decreases by a factor of the batch size.

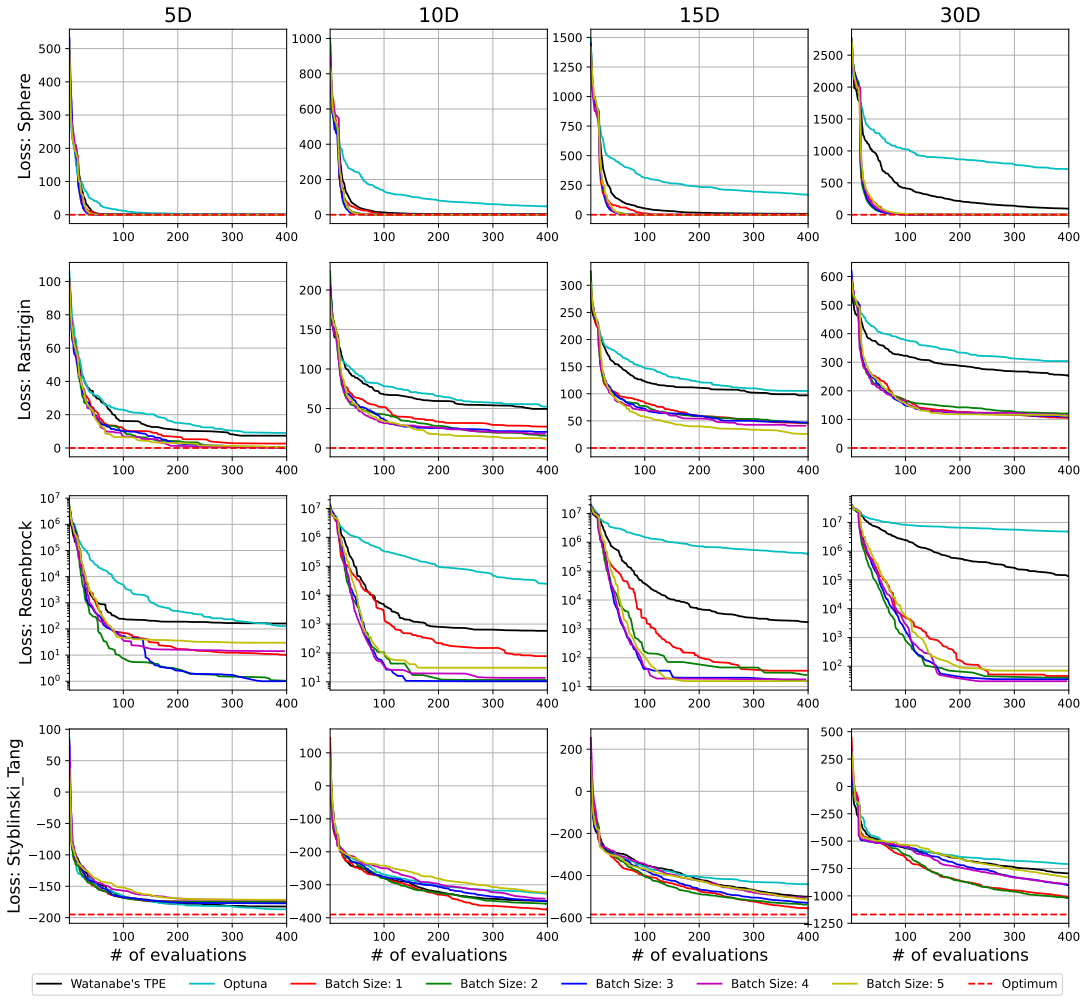


Figure 5.4: The tests on TPE-WLO with the batch size of $\{1, 2, 3, 4, 5\}$. The optimum of Rosenbrock is not shown due to the logarithmic coordinates. Each column has the same dimensions and each row has the same loss function (benchmark).

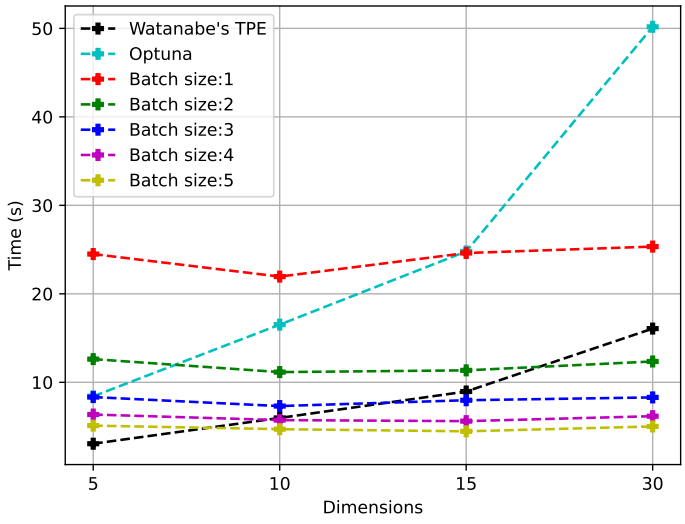


Figure 5.5: The average time of four benchmarks spent on different algorithms.

5.2 Comparison of WLO systems

In this section, the simulation-based system and the hybrid system with TPE-WLO and Watanabe's TPE are tested on three DSP designs, one phase recovery design, and two FIR filters.

5.2.1 Phase Recovery DSP

Börjeson and Larsson-Edefors proposed an FPGA-based fiber emulation system, Fiber-on-Chip (FoC) [8], which can be used to emulate DSP designs including a phase recovery DSP (VV) proposed in [20] (Figure 5.7). Figure 5.6 shows the whole structure of the FoC system. In this test, we use the FoC system[21] to emulate the VV design, where only the random number generator (RNG) and the modulator are used in the transmitter, and in the channel, the additive white Gaussian noise (AWGN) and the phase noise are used. For the VV design, there are five signals to be optimized: the magnitude, the partitioned results, the 2^{nd} power result, the 4^{th} power result, and the phase.

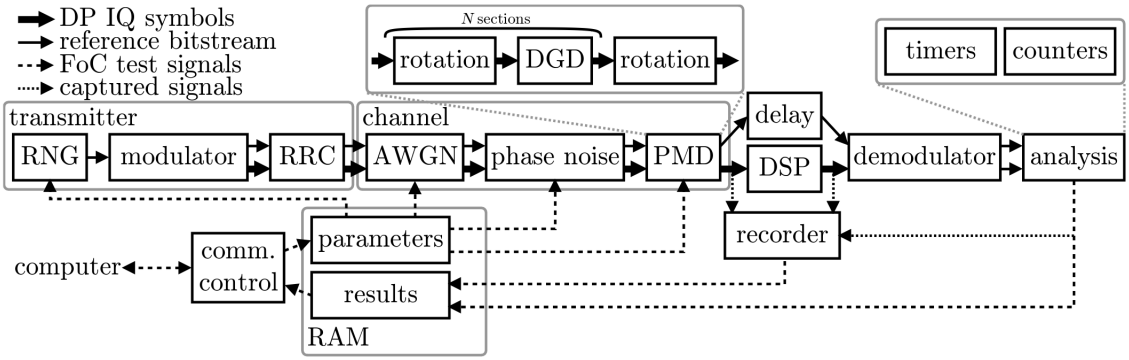


Figure 5.6: The typical block diagram of the Fiber-on-Chip system [8].

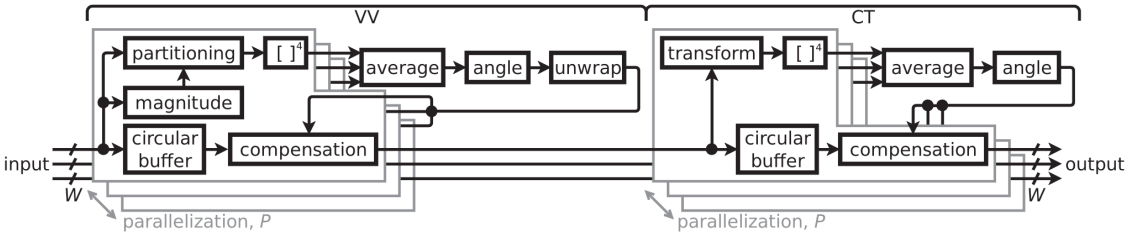


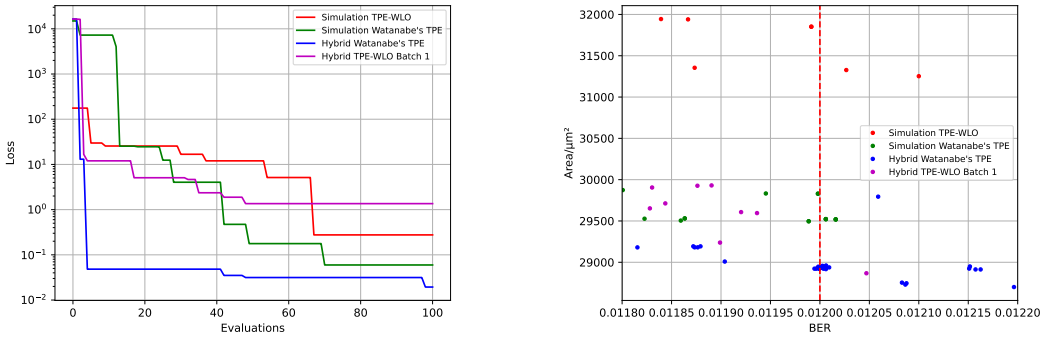
Figure 5.7: The block diagram of the phase recovery DSP [20]. Only the VV part is used.

The metrics used are bit error rate (BER) and area. There are two groups of tests, one is FPGA-emulation-based, and the other is simulation-based. The emulation is carried out on the FoC system with the phase recovery DSP integrated on FPGA, whereas the simulation is carried out on software. Therefore, the differences in the accuracy of other blocks might cause some differences in the results. The key

parameters are set to Table 5.2. It should be noted that due to the unpredictable execution time of our script, the time window list in the table resulted in roughly 1.55 million bits. Therefore, the number of bits in the simulation is set to 1.5 million. In these tests, the accuracy target is set to 0.012, the accuracy range is set to (0.0118, 0.0122)

Table 5.2: Key parameter settings for the FoC system and the phase recovery DSP.

FoC parameters	
Modulation	16 QAM
AWGN coefficient	0x0D6E
Phase noise coefficient	0x0104
Time window length	6e-3 s
Max amplitude	0.5
Phase recovery DSP parameters	
QAM	16
Amplitude scaling	1.0/2.6832
N_SEL_RINGS	2
AVERAGE_LENGTH	64
Other parameters	
SNR	8 dB



(a) Loss function trend **(b)** Area-BER observations

Figure 5.8: Optimization results for the phase recovery DSP design.

Figure 5.8 shows the trend of loss function value as the number of iterations grows and the observation distribution. In both simulation-based system and hybrid system, Watanabe's TPE provided a faster convergence speed and better search results. Due to the different input data for hardware emulation and software simulation, the final results are slightly different. However, the observations of Watanabe's TPE concentrate around the accuracy of 0.012 because the loss function is constructed by the product. The observations of TPE-WLO didn't show the explicit accumulation around the target due to worse exploration. In the simulation-based tests, the time spent is 3612 seconds (16 rounds of synthesis) for Watanabe's TPE and 3207

seconds (7 rounds of synthesis) for TPE-WLO. In the hybrid system, it only takes 3180 seconds (38 rounds of synthesis) for Watanabe's TPE and 2687 seconds (9 rounds of synthesis) for TPE-WLO. It should be noted that, in these tests, the synthesis of the same wordlength configuration is avoided. Overall, Watanabe's TPE can provide faster convergence speed and better solutions in both the simulation-based system and the hybrid system.

5.2.2 FIR filters

The other 2 DSP designs for WLO are transposed finite impulse response (FIR) filters with 15 coefficients (14^{th} order) and 30 coefficients (29^{th} order), which is shown in Figure 5.9 (only the structure of the FIR filter with 15 coefficients is shown due to the similar structure). The metrics used for optimization are MSE and area. In the FIR structure, the multipliers consume the most area. Therefore, the signals to be optimized are the outputs of the multipliers (the EDA tool will automatically remove the unused bits in the multipliers). In the test of 14^{th} order FIR filter, the accuracy target is $4.5 \cdot 10^{-5}$, and the expected accuracy range is set from $3 \cdot 10^{-5}$ to $6 \cdot 10^{-5}$.

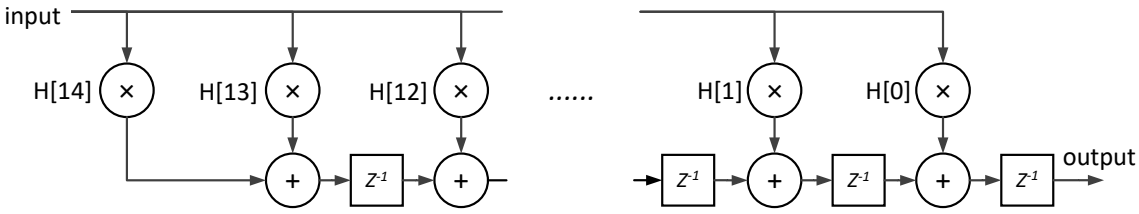
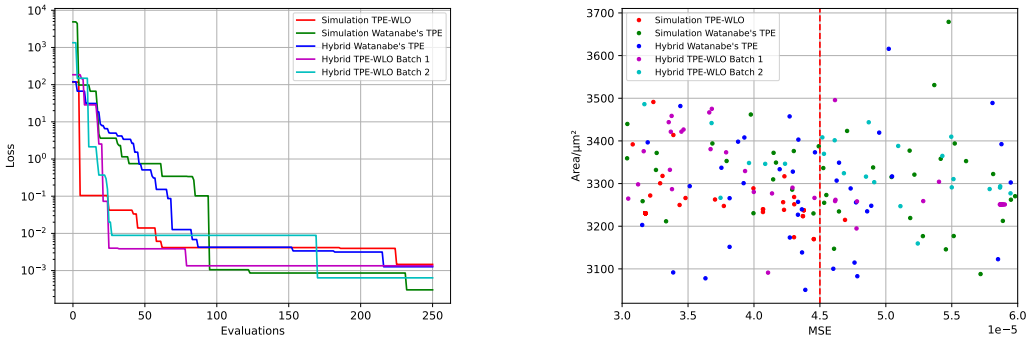


Figure 5.9: The structure of the FIR filter.

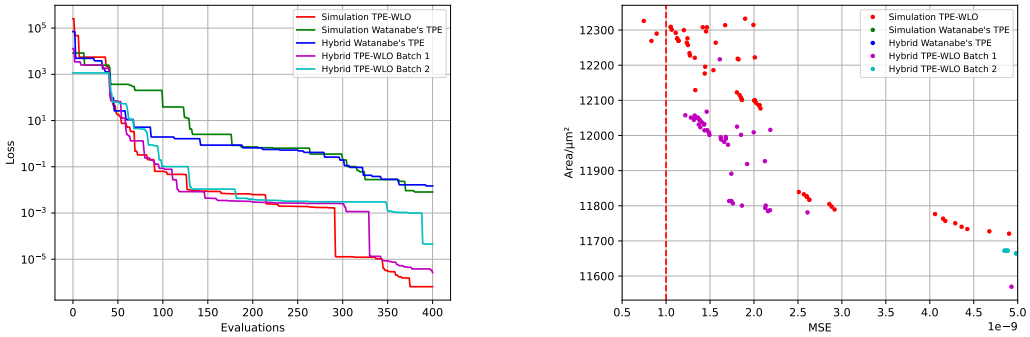


(a) Loss function trend **(b)** Area-MSE observations

Figure 5.10: Optimization results for the 14^{th} order FIR design. The rapid drop in the trends (a) shows the algorithms went into the accuracy range.

Figure 5.10 shows the optimization result of the 14^{th} order FIR. In both simulation-based and hybrid WLO system, our TPE-WLO algorithm has a faster convergence speed in the beginning. But in the following iterations, Watanabe's algorithm can

lead to a better solution (i.e. lower loss). However, the use of our hybrid system has greatly reduced the optimization time. The simulation-based system with Watanabe's algorithm and our TPE-WLO algorithm took 1754 seconds (41 rounds of synthesis, while in other iterations, the synthesis is skipped) and 1673 seconds (54 rounds of synthesis), respectively. And our hybrid system with Watanabe's TPE and the TPE-WLO algorithm only took 1056 seconds (39 rounds of synthesis) and 1129 seconds (37 rounds of synthesis) when the batch size is set to 1, and 894 seconds (24 rounds of synthesis) when using the batch size of 2. The optimization process is expected to be faster when increasing the batch size, but the result does not explicitly illustrate our expectations. The reason is that the random factors in the algorithm part leading to different numbers of synthesis always create variations in the optimization time. On the other hand, from the results of the different batch sizes, a larger batch size produces more points on the right side of the target, i.e. points with a larger MSE, which means the larger batch size could cause a reduction in the exploration of TPE-WLO.



(a) Loss function trend

(b) Area-MSE observations

Figure 5.11: Optimization results for the 29^{th} order FIR filter. In the Area-MSE distributions (b), no points are observed when using Watanabe's TPE.

While in the test of the 29^{th} order FIR filter, Watanabe's TPE fails to converge to the accuracy range, shown in Figure 5.11. The accuracy range is set from $5 \cdot 10^{-10}$, $5 \cdot 10^{-9}$ and the target is $1 \cdot 10^{-9}$. The initial points and total evaluations are set to 40 and 400. The reduction in the exploration due to larger batch size is more explicit in this case. The time for the simulation-based approach with TPE-WLO is 14115 seconds (75 rounds of synthesis). And the time for the hybrid system is 5785 seconds (45 rounds of synthesis) and 484 seconds (8 rounds of synthesis) when the batch sizes are 1 and 2.

5.2.3 Speedup by FPGA acceleration

To quantify the speedup brought by the hardware emulation, the same tests but with the proxy costs are carried out. The proxy cost in this test is the aggregated wordlength, i.e. the sum of all wordlengths. Therefore, the time for obtaining the cost could almost remain the same. The result is presented in Table 5.3, and

demonstrates a reduction in the evaluation time for the accuracy of DSP designs by a factor of 504 for the 29th order FIR filter, 342 for the 14th order FIR filter, and 487 for the phase recovery design, when the batch size is 1. The time spent on simulation is significantly reduced by hardware emulation. However, the further reduction in time by larger batch size is not as great as the ratio of the batch sizes suggests because of the data transmission.

Table 5.3: Time in seconds spent on simulation-based approach and the hybrid system, when the proxy cost is used. The total time for VV (the phase recovery DSP) included the SSH communication delay which made it longer than for the others. The emulation time included the time for emulation and transmission.

Design	Simulation-based		Batch size: 1		Batch size: 2	
	Simulation	Total	Emulation	Total	Emulation	Total
VV	819.01	820.92	1.68	49.87	-	-
14th order FIR	626.61	634.74	1.83	10.06	1.21	5.64
29th order FIR	1760.73	1782.43	3.49	25.36	2.45	13.48

6

Discussion

The discussion on the results and future work are presented in this chapter.

6.1 Discussion on the Results

The discussion on the results mainly refers to limitations and insufficient parts of the tests. The discussion is divided into tests for algorithms and systems.

6.1.1 Results of Algorithms

Stability of the metric

In this project, only crucial parameters in TPE-WLO are tested. Other parameters such as different weighting methods for the surrogate model and the mean point of $D^{(l)}$, different methods for random moves, etc. will also affect the performance. For the parameters tested in the project, only a few of them are evaluated due to project time limitations. The quality of final solutions may differ in different runs even if using the average of 20 runs.

Benchmarks and reference algorithms

The benchmarks used for the evaluation of algorithms are four common mathematical functions adjusted for the integer search space. The results of the reference algorithms on the benchmarks may be a bit different from the results in other literature because of the random feature of the algorithm and the integer search space.

6.1.2 Results of the Hybrid system

Loss function

Although a target is set for the accuracy, the actual optimum of the acquisition function may differ from the accuracy target, because of the impact of the cost. If the cost has values varying more significantly than the accuracy does, the optimum may be greatly different from the target.

6.2 Future Work

The future work from the algorithm part and system part is discussed in this section.

6.2.1 Algorithm

Varying standard deviation

Since the surrogate model of TPE is constructed by Gaussian mixture models, different standard deviations of Gaussian components have different impacts on the quality of the solution in different benchmarks. A simple 1D example of the Gaussian mixture models and the acquisition is shown in Figure 6.1. Larger coefficients can lead to fast convergence due to large search movements, but may cause less exploitation (see red curves in 5D in Figure 5.1). The efficiency of the algorithm depends on the balance of the exploration and the exploitation. Therefore, a varying standard deviation may further improve the performance of the algorithm. A similar study has been carried out by Watanabe [16].

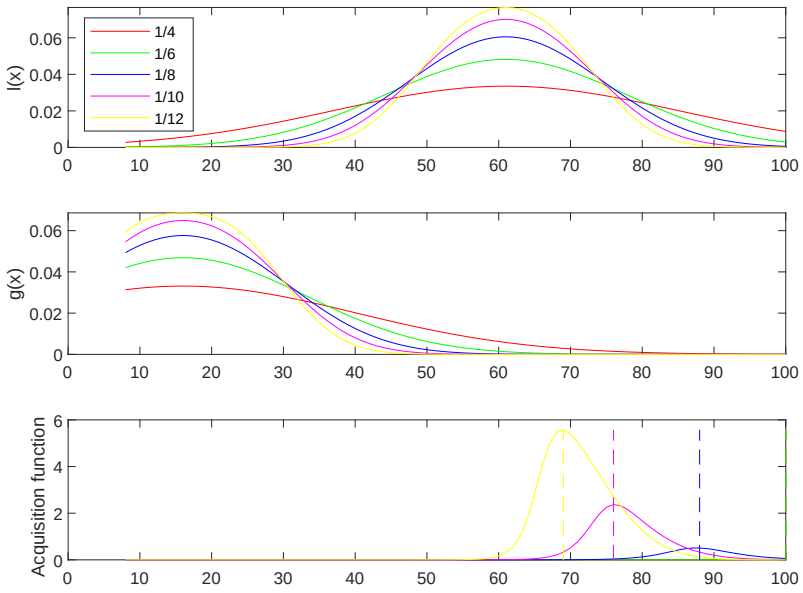


Figure 6.1: A 1D example of different standard coefficients.

Avoid the same configurations

TPE-WLO has detection of the same configuration when adding random moves or generating batch configurations, however, there are still some parts that need to avoid the same configurations. The excessive configurations may not affect the quality of the solution, but will increase the time for evaluation, especially when the design is quite complicated and it takes a long to run synthesis.

Heuristic random move

Fully random moves are used in the algorithm to explore the area near the best configuration when it is in exploration or it detects the same configuration. But

fully random moves are not effective in high dimensions. For example, in 15D, it can be $+1$, -1 or $+0$ for each dimension, so there are 3^{15} random moves. To make moves heuristic, one possible way is analyzing the variation in each dimension of observations in $D^{(l)}$.

Criterion for exploration and exploitation

The only criterion for exploration and exploitation in the algorithm is the value of the empirical function (Equation 4.7). Other criteria such as the stability of several latest samples, proposed in [6] where the author used the stability to stop TPE and enter greed search, may be more effective.

Loss function

As mentioned above, the difference between the optimum and the target depends on the value of the cost. One possible way is to scale the cost or the accuracy by adding a scaling factor as Ha et al. did in [6, 7]. On the other hand, other ways to construct the loss function may also solve the issue, for instance, using addition to combine the metrics.

6.2.2 Hybrid System

More functional accuracy evaluator and input data generator

The accuracy evaluators used in the project are the BER counter for the phase recovery DSP and the MSE calculator for the FIR filter. A more functional accuracy evaluator, such as an SNR calculator, enables the hybrid system to evaluate DSP designs from more perspectives. On the other hand, a more functional input data generator is also necessary to generate reasonable data for different metrics.

High speed transmission interface

Some metrics such as SNR require lots of calculations, making it hard to preprocess the data on hardware. Therefore, transmitting the data to the PC during runtime may be an effective way to solve this. The large amount of data needed for evaluation raises the high demand for transmission speed. In the project, the transmission interface is UART, which is too slow to transmit large amount of data. Ethernet interface may be a better solution.

7

Conclusion

The primary goal of this project is to implement a hardware-accelerated WLO system for DSP designs. To achieve this, we modified the TPE algorithm and developed a hybrid WLO system consisting of the hardware (i.e. FPGA) that emulates DSP designs and the software that carries out the optimization and control. The main features of the design are the fast convergence speed of the algorithm and the acceleration of the evaluation for DSP designs. The fast convergence speed of the algorithm in higher dimensions is presented by the comparisons to the other two reference algorithms, i.e. Watanabe's TPE and Optuna. In addition, three DSP designs were tested by the hybrid WLO system. In the optimization of the phase recovery design (the search space is 5D), Watanabe's TPE presented a better performance in terms of both convergence speed and quality of solution. In the optimization of two FIR filters, our algorithm provided the faster convergence speed, especially in the 29^{th} order FIR filter where Watanabe's TPE failed to converge into the accuracy range.

Although the hybrid WLO system can accelerate the optimization by emulating the DSP design on hardware, the total optimization process still depends on the complexity of the DSP design and suffers from the long synthesis time. A more advanced method is needed to avoid excessive synthesis, which can be considered part of future work. Additionally, a more reasonable combination of metrics in the loss function can also improve the efficiency of the system.

Overall, compared to the conventional simulation-based approach, the long simulation time is greatly reduced by hardware emulation. Our hybrid system can provide higher search efficiency and is flexible to utilize other optimization algorithms.

Bibliography

[1] H. Choi and W. Burleson, "Search-based wordlength optimization for VLSI/DSP synthesis," in *Proceedings of 1994 IEEE Workshop on VLSI Signal Processing*, 1994, pp. 198–207.

[2] M.-A. Cantin, Y. Savaria, D. Prodanos, and P. Lavoie, "An automatic word length determination method," in *IEEE International Symposium on Circuits and Systems*, vol. 5, 2001, pp. 53–56 vol. 5.

[3] H.-N. Nguyen, D. Menard, and O. Sentieys, "Novel algorithms for word-length optimization," in *2011 19th European Signal Processing Conference*, 2011, pp. 1944–1948.

[4] D.-U. Lee, A. Gaffar, R. Cheung, O. Mencer, W. Luk, and G. Constantinides, "Accuracy-Guaranteed Bit-Width Optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 10, pp. 1990–2000, 2006.

[5] J. Hormigo and G. Caffarena, "FPGA acceleration of bit-true simulations for word-length optimization," in *2021 IEEE 28th Symposium on Computer Arithmetic (ARITH)*, 2021, pp. 119–122.

[6] V.-P. Ha and O. Sentieys, "Leveraging Bayesian Optimization to Speed Up Automatic Precision Tuning," in *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2021, pp. 1542–1547.

[7] ——, "Maximizing Computing Accuracy on Resource-Constrained Architectures," in *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2023, pp. 1–6.

[8] E. Börjeson and P. Larsson-Edefors, "Fiber-on-Chip: Digital Emulation of Channel Impairments for Real-Time DSP Evaluation," *Journal of Lightwave Technology*, vol. 41, no. 3, pp. 888–896, 2023.

[9] M.-A. Cantin, Y. Savaria, and P. Lavoie, "A comparison of automatic word length optimization procedures," in *2002 IEEE International Symposium on Circuits and Systems. Proceedings (Cat. No.02CH37353)*, vol. 2, 2002, pp. II–II.

[10] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the Human Out of the Loop: A Review of Bayesian Optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.

[11] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, vol. 13, pp. 455–492, 1998.

[12] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Proceedings of the 24th International Conference on Neural Information Processing Systems*, ser. NIPS'11. Red Hook, NY, USA: Curran Associates Inc., 2011, p. 2546–2554.

[13] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006.

[14] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data Via the EM Algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, 1977. [Online]. Available: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1977.tb01600.x

[15] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.

[16] S. Watanabe, "Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance," 2023. [Online]. Available: https://arxiv.org/abs/2304.11127

[17] V. Vashishtha, M. Vangala, and L. T. Clark, "ASAP7 predictive design kit development and cell design technology co-optimization: Invited paper," in *2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2017, pp. 992–998.

[18] H. Robbins and S. Monro, "A Stochastic Approximation Method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400 – 407, 1951. [Online]. Available: https://doi.org/10.1214/aoms/1177729586

[19] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A Next-Generation Hyperparameter Optimization Framework," in *The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 2623–2631.

[20] E. Börjeson and P. Larsson-Edefors, "Energy-Efficient Implementation of Carrier Phase Recovery for Higher-Order Modulation Formats," *Journal of Lightwave Technology*, vol. 39, no. 2, pp. 505–510, 2021.

[21] CHOICE - Chalmers Optical Fiber Channel Emulator, 2022, www.cse.chalmers.se/research/group/vlsi/choice.