Black-box optimization of parametrically modeled digital circuitry for optical communications

Black-box Optimization of Parametrically Modeled Digital Circuitry for Optical Communications

Tsuyoshi Yoshida[(1)], Hayato Sano[(1)], Shota Koshikawa[(1)], Alifu Xiafukaiti [(1)], Magnus Karlsson[(2)], and Erik Agrell[(2)]

[(1)] Mitsubishi Electric Corporation, Japan, Yoshida.Tsuyoshi@ah.MitsubishiElectric.co.jp
[(2)] Chalmers University of Technology, Sweden

Abstract *An efficient development method is required for large-scale digital signal processing implementations. The proposed design method with parametric modeling and multi-objective optimization reduces the optimization time from 300 years by brute-force search to around 2 weeks by a heuristic solver using approximation and machine learning. ©2024 The Author(s)*

Introduction

Model-based design (MBD) and model-based systems engineering are general tools for constructing reliable cyber-physical systems [1–3]. Such systems consist of processing units including a central or graphics processing unit, a field-programmable gate array (FPGA), an artificial intelligence core [4], etc. The MBD process must be continuously improved to keep up with the ever-increasing system scale and decreasing delivery times [5]. Model-based machine learning (ML) methods have been applied for, e.g., constellation optimization in optical communications [6]. High-performance optical communications are supported by digital signal processing (DSP) implemented in large-scale integrated (LSI) circuitry [7], which requires an efficient development method as well as general processing units.

MBD includes *auto-coding,* which converts a model into raw codes describing operations, avoiding manual coding efforts. On the other hand, extensive manual work is required whenever the *implementable model,* i.e., the model fed into the auto-coder, is revised, even if there is only a minor change in the model parameters. Furthermore, existing hardware codes are rendered useless when the model is revised.

Systems are characterized by *functional metrics* such as error vector magnitude (EVM) and bit error rate, and *implementational metrics* such as resource utilization, power consumption, and timing margin [8,9]. The multi-objective optimization of parameter combinations depends on the system requirements. The resulting metrics of both types are estimated through the black-box behavior of the auto-coder and the synthesis and layout tools.

Even with auto-coding, it is complicated to design a good system with processing units because of the usual development flow, i.e., first choosing upper-layer parameters, then constructing a specific implementable model, converting the model to raw codes, and finally feeding the codes to synthesis and layout tools. If one stage fails, we must revisit previous stages in the flow. This iterative process makes the development inefficient.

To fully enjoy the benefit of MBD, the implementable model should be a parametric one, where variable parameters enable the model to be reconfigured for various concrete use cases without remodeling. Once a parametric model is given, the parameters can be determined by black-box multi-objective optimization.

In this work, we propose parametric and implementable modeling of digital circuitry, combined with black-box optimization of the model parameters according to the system requirements. As a proof of concept, the framework is successfully applied to DSP design in a coherent fiber-optical receiver, implemented on an FPGA.

Modeling method

Fig. 1 shows flowcharts of development methods. In the conventional method (a), the upper-layer design fixes the functions and parameters based on system requirements, followed by a specific modeling stage, which outputs an implementable model (I-model). Then, auto-coding converts the I-model to raw hardware description language (HDL) codes, and synthesis and layout are performed to obtain the final circuitry specifications for implementation on a target processing unit. The required tools and expertise usually differ significantly between the upper-layer algorithmic design and the lower-layer implementation tasks,

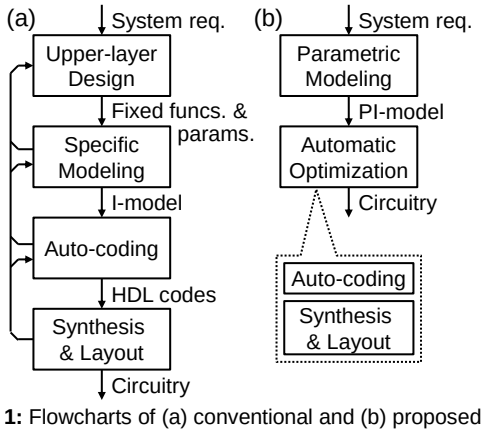

Fig. 1: Flowcharts of (a) conventional and (b) proposed development methods.

which makes the determination of both algorithmic and implementational parameters an iterative process. On the other hand, in the proposed simplified flowchart (b), a parametric and implementable model (PI-model) is constructed, which can be directly converted to a concrete model by assigning each parameter a specific value. Based on the PI-model, all algorithmic and implementation parameters are determined by automatic optimization. The final circuitry specification is obtained through auto-coding, synthesis, and layout (ACSL), which are iteratively performed inside the automatic optimization step. In this work, we utilize Simulink as a modeling tool, taking care to avoid the predefined models in Simulink that cannot be converted to HDL codes.

In our trial, the pilot-aided DSP chain in coherent reception for 4, 16, 64, 256-ary quadrature amplitude modulation (QAM) [10,11] is exemplified, whose block diagram is shown in Fig. 2. The eight functions F1–F8 are optimization targets and two external function blocks (dashed) serve to evaluate the communication quality. The input-side external functions include an optical transmitter for 2 subcarrier-multiplexed polarization-division-multiplexed 16-QAM at 2 Gsymbol/s, an optical channel with additive noise at a signal-to-noise ratio of 50 dB, and analog-to-digital conversion at 5 Gsample/s. The subcarrier spacing was 2.4 GHz, the carrier frequency offset was 50 MHz, and the laser linewidth was ~10 kHz. The receiver (Rx)-side state controller monitors the clock, frequency, and phase synchronization states and controls the corresponding functions. The output-side external functions include frame synchronization and EVM calculation. In each function, the number of bits defining the amplitude resolution and the insertion density of delay flip-flops (usually put around complex processing such as multiplications) are described with variables. Both static and adaptive equalization are implemented by finite-impulse-response filters, where the number of taps and the number of parallel input symbols are described with variables.

The parameters to be automatically optimized are limited to the number of taps in F1 ($p_1 \in \{36, 40, \ldots, 72\}$) and the numbers of bits for amplitude levels in F1, F3, F5, F6, and F7 ($p_2, \ldots, p_6 \in \{10, 11, \ldots, 19\}$). These 6 variables with 10 cases each yield up to 10^6 possible combinations.

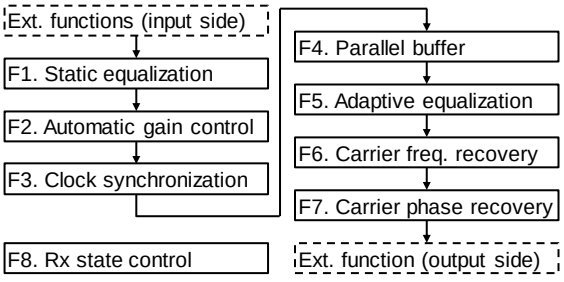

Fig. 2: Exemplified block diagram with parametric modeling.

Optimization method

Even if we construct a PI-model and put the parameter determination into the hands of an optimizer, the time to explore solution spaces based on the model simulation and ACSL can be prohibitive. For example, a full model simulation costs 3 minutes, a full model ACSL takes 3 hours, and the number of solution candidates would usually be >10000 based on the PI-model. To obtain a reasonable solution within a limited time, the number of candidates or the time per candidate needs to be reduced. To address this issue, first, we reduce the ACSL time by breaking the full model into component models and combining each ACSL results to approximate the full model ACSL. Second, we introduce a factorization machine with annealing (FMA) [12–15], which is a heuristic solver using ML. The annealing optimizer minimizes the energy in systems described by the *Ising model,* a physics-based model with binary variables [16–19].

Fig. 3 shows the flowchart of the proposed optimization method. In step S1, reference information is gathered to define the conditions for the optimization and approximation in S2. The ACSL with component models in S3 outputs component-wise circuit qualities, which are converted to system-wide quality metrics according to references in S1. These are used in S4 to obtain the full model's estimated circuitry cost $\hat{C}_\mathrm{c}$ for all considered cases. The key optimization step S5 extracts N potential solution candidates expected to minimize the estimated total cost $\hat{C}_\mathrm{t} = \hat{C}_\mathrm{c} + C_\mathrm{f}$, where C_f denotes the function cost obtained from model simulations with Simulink. Step S6 performs ACSL based on full system models to obtain the actual circuitry cost C_c for the cases extracted in S5. Since C_c can differ from $\hat{C}_\mathrm{c}$, there may be multiple solution candidates. Step S7, finally, chooses a solution from the candidates or

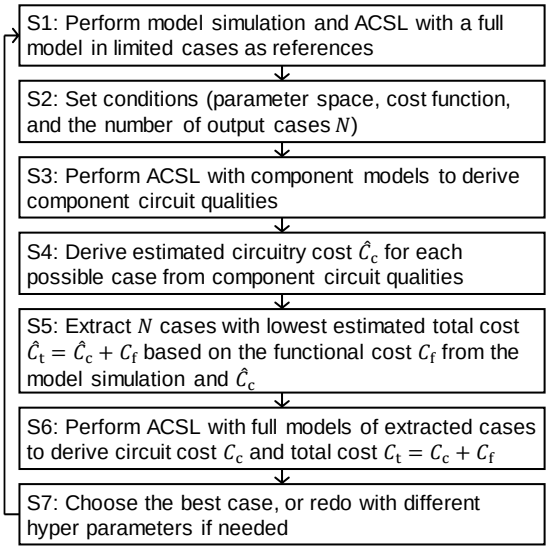

Fig. 3: Flowchart of the proposed optimization method, where ACSL denotes auto-coding, synthesis and layout.

initiates another iteration with different conditions.

Both C_c and $\hat{C}_c$ are obtained from ACSL using Simulink for the auto-coding and Xilinx Vivado for the synthesis and layout. Step S5 employs FMA, whose acquisition function is

$$g(\boldsymbol{x},\boldsymbol{w}) = w_0 + \langle \boldsymbol{x},\boldsymbol{w}\rangle + \sum_{i<j}\langle \boldsymbol{v}_i,\boldsymbol{v}_j\rangle x_i x_j, \quad (1)$$

where $\boldsymbol{x}$ denotes the feature vector with elements $x_i \in \{0,1\}$, w_0 the global bias, $\boldsymbol{w}$ the linear weights, and $\boldsymbol{v}$ the latent vector. The second and third terms in the r.h.s. of (1) correspond to the magnetic field and the coupling coefficient in an annealer, respectively. The parameter combinations to be considered are encoded into $\boldsymbol{x}$. The ML determines $\boldsymbol{w}$ and $\boldsymbol{v}$, and an annealing/Ising solver then finds $\boldsymbol{x}$ to minimize (1).

Demonstration

Based on the full PI-model with component models Fk for k=1, …, 8 and parameters p_1, …, p_6 in Fig. 2, the black-box multi-objective optimization process in Fig. 3 was performed. The considered metrics l=1, …, 7 were 1) EVM as a functional quality, 2) the look-up table (LUT) size, 3) register size, 4) DSP size, 5) worst negative slack (WNS), 6) worst hold slack (WHS), and 7) power consumption. The full model's value of metric l is $\chi_l = \sum_k \chi_{kl}$ for l=1, 2, 3, 4, 7 and $\chi_l = \min_k \chi_{kl}$ for l=5, 6, where χ_{kl} is metric l of component model k. Differences between the metric values for the component models and the full model were compensated by linear regression, i.e., $\chi'_l = a_l\chi_l + b_l$ based on the results in S1 and S4. Defining the low-cost limit α_l, the acceptable limit β_l, the cost weight ω_l, and the maximum cost M_l, the estimated elemental costs for l=1, 2, 3, 4, 7 are $\hat{C}_l = \max\left(0, \omega_l(\chi'_l - \alpha_l)/(\beta_l - \alpha_l)\right)$ if $\chi'_l \le \beta_l$ and $\hat{C}_l = \omega_l M_l$ otherwise. The ones or for l=5, 6 are $\hat{C}_l = \max\left(0, \omega_l(\alpha_l - \chi'_l)/\alpha_l\right)$ if $\beta_l \le \chi'_l$ and $\hat{C}_l = \omega_l M_l$ otherwise. The estimated total cost $\hat{C}_t = \sum_l \hat{C}_l$.

We followed the flowchart in Fig. 3, where S1 gathered reference data in 48 cases, considering Xilinx Zynq ZCU208 as the target device. S2 determined the parameters defining the problem (ω_l, α_l, and β_l) and approximating the metrics (a_l and b_l) as shown in Tab. 1, where M_l=100 for every l and N=20 in S5. In S3, there were 100 cases for F1, 10 cases for F3, F5, F6, and F7, and 1 case for F2, F4, and F8; 143 cases in total because of the component-wise ACSL. S4 computed $\hat{C}_c$ for 10^6 cases. S5 employed FMA with energy

$$E[\boldsymbol{x},m] = -1/\left|1 - \hat{C}_b[\boldsymbol{x},m]/\hat{C}_t[\boldsymbol{x},m]\right|^d, \quad (2)$$

where m denotes the cumulative number of model simulations, $\hat{C}_b$ the estimated boundary value $< \hat{C}_t$, and d a positive real value. Here, $\hat{C}_b[\cdot,m] = 0.9 \cdot \min\left(\hat{C}_t[\cdot,m-1]\right) \ge 0$ and d=1. The size of $\boldsymbol{v}$ was set to 8. Fig. 4 shows the results of black-box optimization. The first ML was performed from the dataset of initial sampling of $\boldsymbol{x}$ and the corresponding E (filled circles in Fig. 4) in (2) with n times model simulations. The annealer found $\boldsymbol{x}$ in (1) as the next sample and (2) calculated its E. When each additional pair of $\boldsymbol{x}$ and E were obtained, the ML and annealing were iteratively performed (open circles in Fig. 4). The model simulations in S5 to obtain C_f dominated the processing time. The lowest costs $\hat{C}_t$ for n=800, 400, and 200 were 3.04, 2.95, and 2.93, respectively. In this trial, n=400 in Fig. 4 was efficient in obtaining many samples having a low energy or cost with a small m. According to N=20 cases chosen with the estimated cost $\hat{C}_t$ in S5, S6 derived the actual cost C_t (crosses in Fig. 4). While the actual timing margins obtained in S6 are hard to estimate from the component models in S5, any nonnegative values are acceptable, where unacceptable cases can appear in S6. The other metrics in S6 agreed with the ones estimated in S5. The lowest C_t of 2.97 was obtained with parameters p_1, …, p_6=36, 15, 13, 11, 18, 12.

The processing time for each step were approximately 150 hours in S1 (48 cases), 100 hours in S3 (143 cases), 30–50 hours in S5 (600–1000 cases), 60 hours in S6 (20 cases), and 340–360 hours (about 2 weeks) in total.

Tab. 1: Parameters defining and approximating the problem.

l	metric	ω_l	α_l	β_l	a_l	b_l
1	EVM	4.5	10	5.0	1	0
2	LUT	1.0	4.0e5	7.5e4	2.5	–3.2e5
3	Regs.	1.0	8.0e5	2.5e4	0.78	–1.0e4
4	DSP	1.0	8.0e3	2.0e3	0.059	3.6e4
5	WNS	0.25	0	0.1	1	0
6	WHS	0.25	0	0.05	1	0
7	Power	1.0	20	10	1.9	–38

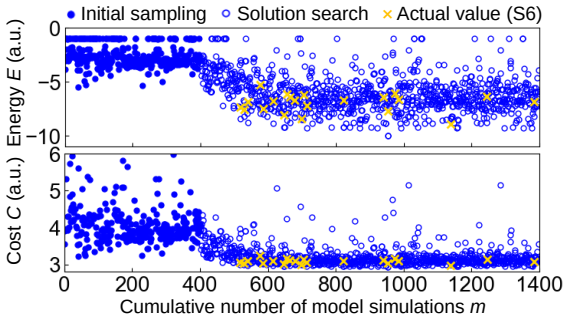

Fig. 4: Results of black-box optimization ($\boldsymbol{n}$=400): circles are estimated values in S5 and crosses are actual values in S6.

Conclusions

We proposed the concept of parametric modeling with automatic optimization and applied to the design of the DSP chain on an FPGA in a coherent receiver. While brute-force optimization with the full model would require >300 years at maximum, the proposed heuristic solver with the approximation and FMA reduced the time to around 2 weeks.

Potential future works could be an application of Bayesian optimization [20,21] and the use of large language models for generating HDL codes directly from the system requirements.

Acknowledgements

This work was in part supported by the commissioned research of National Institute of Information and Communications Technology (NICT), Japan, under grant numbers JPJ012368C01401 and JPJ012368C08401.

References

[1] J. C. Jensen, D. H. Chang and E. A. Lee, “A model-based design methodology for cyber-physical systems,” in Proc. International Wireless Communications and Mobile Computing Conference, Istanbul, Turkey, 2011, pp. 1666–1671, doi: 10.1109/IWCMC.2011.5982785.

[2] P. Derler, E. A. Lee, and A. S. Vincentelli, "Modeling cyber–physical systems," in Proc. IEEE, vol. 100, no. 1, pp. 13–28, 2012, doi: 10.1109/JPROC.2011.2160929.

[3] L. Li, N. L. Soskin, A. Jbara, M. Karpel and D. Dori, “Model-based systems engineering for aircraft design with dynamic landing constraints using object-process methodology," IEEE Access, vol. 7, pp. 61494–61511, 2019, doi: 10.1109/ACCESS.2019.2915917.

[4] E. Nurvitadhi, D. Sheffield, Jaewoong Sim, A. Mishra, G. Venkatesh and D. Marr, “Accelerating binarized neural networks: Comparison of FPGA, CPU, GPU, and ASIC,” in Proc. International Conference on Field-Programmable Technology (FPT), Xi'an, China, 2016, pp. 77–84, doi: 10.1109/FPT.2016.7929192.

[5] R. Nane et al., "A survey and evaluation of FPGA high-level synthesis tools," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 35, no. 10, pp. 1591–1604, 2016, doi: 10.1109/TCAD.2015.2513673.

[6] V. Neskorniuk, A. Carnio, D. Marsella, S. K. Turitsyn, J. E. Prilepsky and V. Aref, “Model-Based Deep Learning of Joint Probabilistic and Geometric Shaping for Optical Communication,” in Proc. Conference on Lasers and Electro-Optics (CLEO), San Jose, CA, USA, 2022, Paper SW4E.5, doi: 10.1364/CLEO_SI.2022.SW4E.5.

[7] H. Sun, M. Torbatian, M. Karimi, R. Maher, S. Thomson, M. Tehrani, Y. Gao, A. Kumpera, G. Soliman, A. Kakkar, M. Osman, Z. A. El-Sahn, C. Doggart, W. Hou, S. Sutarwala, Y. Wu, M. R. Chitgarha, V. Lal, H.-S. Tsai, S. Corzine, J. Zhang, J. Osenbach, S. Buggaveeti, Z. Morbi, M. I. Olmedo, I. Leung, X. Xu, P. Samra, V. Dominic, S. Sanders, M. Ziari, A. Napoli, B. Spinnler, K.-T. Wu, and P. Kandappan, “800G DSP ASIC Design Using Probabilistic Shaping and Digital Sub-Carrier Multiplexing,” Journal of Lightwave Technology, vol. 38, no. 17, pp. 4744–4756, 2020, doi: 10.1109/JLT.2020.2996188.

[8] 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, doi: 10.1109/JLT.2020.3027781.

[9] E. Börjeson, E. Deriushkina, M. Mazur, M. Karlsson, and P. Larsson-Edefors, “Circuit implementation of pilot-based dynamic MIMO equalization for coupled-core fibers,” in Proc. Optical Fiber Communication Conference (OFC), San Diego, CA, USA, 2024, Paper W1E.4.

[10] M. Mazur, J. Schröder, A. Lorences-Riesgo, T. Yoshida, M. Karlsson, and P. A. Andrekson, “Overhead-optimization of pilot-based digital signal processing for flexible high spectral efficiency transmission,” Optics Express, vol. 27, no. 17, pp. 24654–24669, 2019, doi: 10.1364/OE.27.024654.

[11] K. Matsuda, H. Sano, Y. Takada, M. Binkai, S. Koshikawa, Y. Yokomura, T. Yoshida, Y. Konishi, and N. Suzuki, “Multi-aperture transmission and DSP technique for beyond-10 Tb/s FSO networks,” IEEE International Conference on Space Optical Systems and Applications (ICSOS), Virtual, Japan, 2022, pp. 236–239, doi: 10.1109/ICSOS53063.2022.9749745.

[12] S. Rendle, “Factorization machines,” in Proc. IEEE International Conference on Data Mining, Sydney, NSW, Australia, 2010, pp. 995–1000, doi: 10.1109/ICDM.2010.127.

[13] K. Kitai, J. Guo, S. Ju, S. Tanaka, K. Tsuda, J. Shiomi, and R. Tamura, “Designing metamaterials with quantum annealing and factorization machines,” Physical Review Research, vol. 2, no. 1, pp. 013319-1–10, 2020, doi: 10.1103/PhysRevResearch.2.013319.

[14] Y. Seki, R. Tamura, and S. Tanaka, “Black-box optimization for integer-variable problems using Ising machines and factorization machines,” arXiv:2209.01016, 2022, doi: 10.48550/arXiv.2209.01016.

[15] T. Inoue, Y. Seki, S. Tanaka, N. Togawa, K. Ishizaki, and S. Noda, “Towards optimization of photonic-crystal surface-emitting lasers via quantum annealing,” Optics Express, vol. 30, no. 24, pp. 43503–43512, 2022, doi: 10.1364/OE.476839.

[16] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, “Equation of state calculations by fast computing machines,” Journal of Chemical Physics, vol. 21, no. 6, pp. 1087–1092, 1953, doi: 10.1063/1.1699114.

[17] S. Kirkpatrick, C. D. Gelatt, Jr, and M. P. Vecchi, “Optimization by simulated annealing,” Readings in Computer Vision, vol. 220, no. 4598, pp. 606–615, 1983, doi: 10.1126/science.220.4598.67.

[18] T. Kadowaki and H. Nishimori, “Quantum annealing in the transverse Ising model,” Physical Review E, vol. 58, no. 5, pp. 5355–5363, 1998, doi: 10.1103/PhysRevE.58.5355.

[19] D-wave, The Advantage™ Quantum Computer, [Online]. Available: www.dwavesys.com/solutions-and-products/systems/

[20] J. Močkus, “On Bayesian methods for seeking the extremum,” Optimization Techniques IFIP Technical Conference Novosibirsk, 1974. Lecture Notes in Computer Science, vol. 27, Springer, Berlin, Heidelberg, pp. 400–404, 2005, doi: 10.1007/3-540-07165-2_55.

[21] 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,” in Proc. IEEE, vol. 104, no. 1, pp. 148–175, 2016, doi: 10.1109/JPROC.2015.2494218.