The 23rd International Technical Conference on Circuits/Systems,
Computers and Communications (ITC-CSCC 2008)

# Arithmetic Circuit Optimization in Finite Word Length Approximation of Arbitrary Functions

Takao Sasaki[1], Koji Kotani[1] and Hisamishi Toyoshima[1]
[1]Faculty of Engineering, Kanagawa University
3-27-1 Rokkakubashi, Kanagawa-ku, Yokohama, Kanagawa, 221-8686, Japan
Email: sasaki@tysm.ee.kanagawa-u.ac.jp

**Abstract**: In digital circuit design using a hardware description language, some elementary functions and user defined functions can not be expressed directly. In such cases, the target function is approximated as a polynomial to be implemented by additions and multiplications. If an approximation error exceeds the permissible one, it can be corrected with a look-up table (LUT). However, considering discrete and nonlinear relation between the approximation error and the circuit area, optimization of reducing the total hardware cost would be more complex. In this research, for an arbitrary function, we propose the optimization technique of minimizing the hardware cost. The proposed method starts with multiple initial solutions based on two hardware models. Furthermore, for optimization algorithm, tabu search (TS) is used, and it is parallelized to make a global search.

*Keywords*— **finite word length approximation, coefficient quantization, optimization, tabu search**

## 1. Introduction

Recently, in digital circuit design, an operation specification is provided using software algorithm like a C language. However, in design of some elementary functions and user defined functions, it is necessary to convert from software description to hardware description using hardware description language, ex.) VHDL and Verilog HDL. For example, sine function is described by "sin" in the software description, but in the hardware description, such function is described by the polynomial using adder, multiplier and bit shift, and/or the LUT using ROM. For functions as general as the elementary function, the hardware description method is well known. However, for user defined functions, there is no general method of conversion to hardware description.

In general, arbitrary functions designed by the polynomial are represented by the Chebyshev's approximation. However, in representation of the polynomial using the Chebyshev's approximation, coefficients need infinite word length. In the other case of using the LUT [1], to store all the output values of function, many amounts of the memory are needed. In both case, the hardware cost is very high. Then, to reduce the hardware cost, the target function is approximated as the polynomial using finite word length to be implemented by additions and multiplications, and the approximation error exceeds the permissible one, it can be corrected with the LUT [2]. However, relationship of reducing of the hardware cost of the arithmetic circuit and that of the LUT are trade-off. In addition, considering the discrete and nonlinearity relation between the coefficient quantization and the hardware cost of the LUT, optimization of the total hardware cost would be more complex.

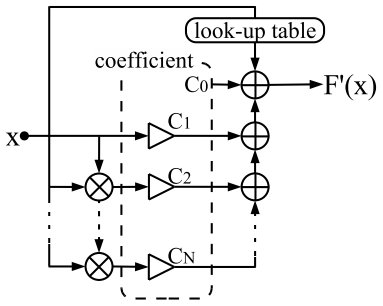


Figure 1. Block diagram of the polynomial representation

In this research, for an arbitrary function, we propose the optimization technique of selecting the appropriate hardware model with less quantization error, and minimizing the hardware cost using the TS. The proposed method starts with multiple initial solutions based on two hardware models, and the TS is parallelized to makes a global search.

## 2. Circuit Representations

In digital circuit design, an arbitrary function is represented as any approximation using simple operations such as multiplications, additions and shifts. For hardware model of the arbitrary function representation, we employ two type models, the polynomial representation and a factorized representation.

### 2.1 Polynomial Representation

In general, the polynomial representation is used for the approximation of the function. The target function $F(x)$ is approximated as an $N$-$th$ polynomial, $F'(x)$ is expressed as Equation 1 and Figure 1,

$$F'(x) = \sum_{i=0}^{N} C_i x^i \tag{1}$$

where each $C_i$ is a coefficient of the polynomial. In this method, the Chebyshev's approximation using coefficients with infinite word length is effective. However, actual circuits using coefficients with finite word length

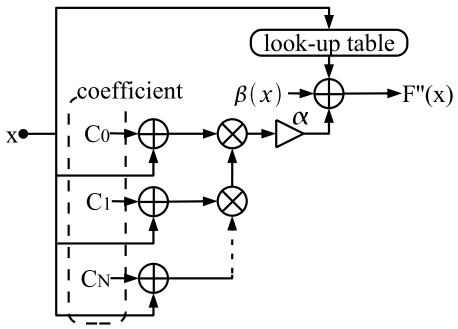


Figure 2. Block diagram of the factorized representation

have the quantization error. To correct such error, the LUT is used.

### 2.2 Factorized Representation

For the approximation of some functions with oscillation component, the factorized representation is effective. The polynomial $F'(x)$ is transformed into the factorized form as

$$F''(x) = \alpha \prod_{i=0}^{N} (x - C_i) + \beta(x) \qquad (2)$$

where $\alpha$ and $\beta(x)$ are scale factor and linear element, respectively. In this representation, the value of the function oscillates with respect to $\beta(x)$. When $x = C_i$, approximated function crosses to the $\beta(x)$, then divergence of this function is prevented. A schematic view of the factorized representation circuit is shown as Figure 2. The quantization error are corrected by the LUT, same as the polynomial representation.

## 3. Optimization Method

As described above, arithmetic circuits are consisted of the approximation circuits and the LUT. The hardware cost of the LUT depends on a word length of the quantization error. However, to reduce the quantization error, large word length of coefficients of the approximation function is necessary. Therefore, the quantization error and the hardware cost of the LUT are trade-off. Then, it is difficult to reduce the hardware cost of the approximation circuit and the LUT simultaneously.

In this research, to reduce the total hardware cost of the arithmetic circuit is dealt with combinatorial optimization problem. To solve such problem, genetic algorithm, simulated annealing, TS and so [3] on are well known. In this problem, coefficients obtained from the Chebyshev's approximation are used as initial solution of optimization. The TS [3]-[5] is useful tool for local search. Then, selection of hardware model and coefficients of approximated function are optimized using the TS.

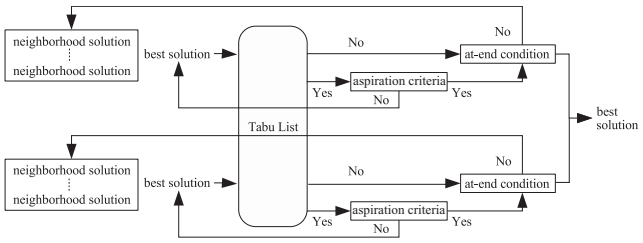


Figure 3. Flowchart of parallel tabu search

### 3.1 Parallel Tabu Search

The TS is a meta-heuristic algorithm that can be used for solving the combinatorial optimization problems proposed by Fred Glover [4][5], and it is belonging to the class of local search techniques. The feature of the TS is cycling back to previously visited solutions is prevented by the use of memories, called tabu lists, that record the recent history of the search.

Since the TS depends on initial solution, initial value setting are very important. In this research, the TS is parallelized with multi search points and probabilistic selection is added to spread a search area using plural initial solutions. So the parallel TS (PTS) have plural initial solutions. Main initial solutions of the PTS are obtained from coefficients of the polynomial using the Chebyshev's approximation and the factorized form. Each coefficient is rounded in finite word length. In addition to these, 4 initial solutions are used. These are summarized below.

(1) all coefficients are zero
(2) all coefficients are maximum value
(3) all coefficients are minimum value
(4) all coefficients are given at random

In the case of (1), the arithmetic circuit is not use, namely the target function is composed of only the LUT. By this solution, the circuit designed only with the table is obtained in worst case. (2) and (3) are edge of solution space, and these cover the solution space widely. (4) is irregular and the solution that cannot be expected quite can be obtained. These supply another possibility that can not obtain from rounded coefficients of the Chebyshev's approximation.

Furthermore, in this research, the tabu list is shared by all solutions of the PTS, since the relation of each solution is deep. A flowchart of this method is shown in Figure 3.

### 3.2 Solution Representation

Solutions of the PTS are represented as a sequence of all coefficients. The format of each coefficient is one sign digit and decimal fraction expressed by binary. $N$ coefficients are aligned in ascending order, for the polynomial form. In the factorized form, the scale factor $\alpha$ is added. Since the total word length of solution is same, the degree of the factorized is smaller than the polynomial. Example of solution representation of the polynomial and the

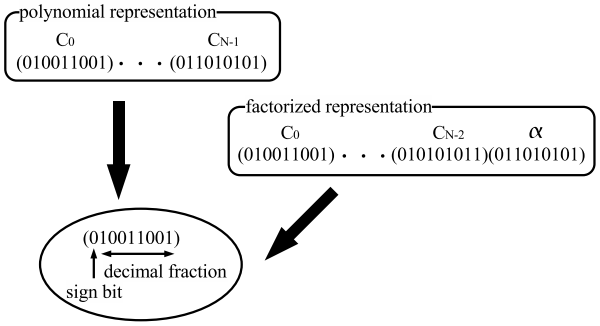


Figure 4. Solution representation of the polynomial and the factorized form

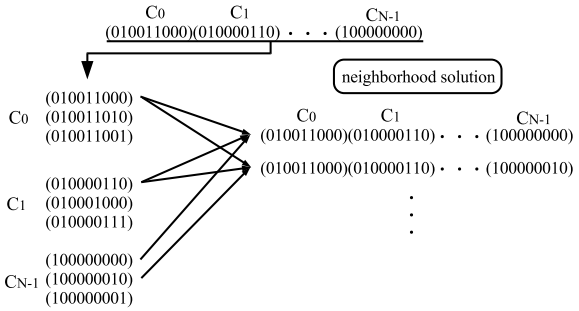


Figure 5. Generation of neighborhoods

factorized form are shown in Figure 4.

The neighborhood solution is generated by changing the least significant bit of coefficients. All coefficients do not generate neighborhood solution because of convergence of neighborhood solution and increase of computation time. The changing coefficients are selected at random using short-term and long-term memory. The sort-term memory registers the number of chance not to have been selected as candidate for each coefficient, and long-term memory has accumulation number in which neighborhood is generated. By using these memories, the coefficient with a little selected frequency is selected by priority. Figure 5 shows an example of generation of neighborhoods, when coefficients $C_0$, $C_1$ and $C_{N-1}$ are selected.

### 3.3 Evaluation

In this research, solutions are evaluated by the hardware cost of the arithmetic circuit and the LUT. The cost of the LUT depends on the word length of maximum value of an error ($le$) between the target function and the approximation. $le$ can be paraphrased as an input word length of the LUT. The dependency of the hardware cost of the LUT on the input word length is shown in Figure 6. The hardware cost of the arithmetic circuit is evaluated by number of adders that compose the circuit. The number of adders is obtained from the number of nonzero bit of each coefficient. In this research, coefficients are expressed as canonical signed-digit (CSD) [6]. CSD representation has fewest nonzero digits and can be efficiently realized by using a small number of adders. The cost of the LUT ($C_{table}$) is expressed by below equation.

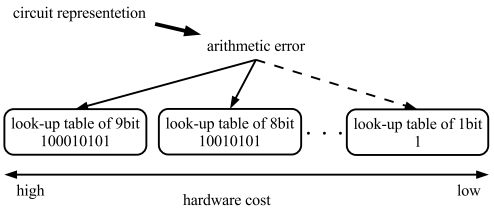


Figure 6. Relationship between the input word length and the hardware cost of the LUT

$$C_{table} = 2^{le} \tag{3}$$

The hardware cost of the LUT is evaluated using $C_{table}$, but evaluation of the total hardware cost needs a consideration of reduction of the cost of the arithmetic circuit. Then, the total hardware cost is evaluated by

$$EV = \frac{1}{C_{table}} + \frac{1}{C_{table} \times C_{adders}}, \tag{4}$$

where $C_{adders}$ is the number of adders of the arithmetic circuit, and the evaluation value is so good that the value is large.

## 4. Simulation Results

To show the effectiveness of the proposed method, design simulations of two typical arbitrary functions have been achieved. Each target function is a combination of polynomials and elementary functions with 9-bit word length of input/output. The specification of these target functions $F_1(x)$ and $F_2(x)$ are shown in Figure 7 and 8, respectively. Design parameters used in this simulation of each example are shown below.

TS:
- # of tabu list : 7
- # of changing coefficients : 3
- # of neighborhood : 27
- # of iteration : 20000
- # of parallels : 12

Target Functions:
- $F_1(x) = \cos(x)\log(x+1)$
- $F_2(x) = (x+1)\log(x+1)$

- Degree N : 8
- Range of $x$ : $0 \leq x \leq 8$

Since $F_1(x)$ has oscillation component, the factorized form is used, and $F_2(x)$ is optimized using the polynomial representation. Table 1 shows these simulation results using the proposed method together with the comparison with the circuit obtained by the Chebyshev's approximation with quantized coefficients of 9-bit.

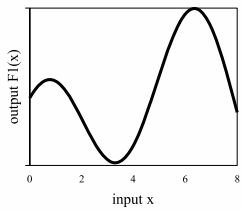


Figure 7. Specification of function $F_1(x)$

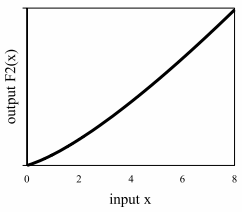


Figure 8. Specification of function $F_2(x)$

Table 1. Comparison of the hardware cost between proposed method and the Chebyshev's approximation with finite word length coefficients

| function | $F_1 : \cos(x)\log(x+1)$ | |
|---|---|---|
| | proposed method | Chebyshev |
| # of adders | 15 | 19 |
| table size | 1 | 8 |
| function | $F_2 : (x+1)\log(x+1)$ | |
| | proposed method | Chebyshev |
| # of adders | 11 | 11 |
| table size | 5 | 8 |

As seen from these results, function $F_1(x)$ is designed both with reduction in the number of adders by 20% and with reduction in the size of the LUT by 90%. For function $F_2(x)$, the number of adders is same, but the size of the LUT reduce to about 60%. These results show that the total hardware cost obtained from proposed method is smaller than that of the Chebyshev's approximation with finite word length coefficients for typical arbitrary functions both with and without oscillation component.

## 5. Summary

In this research, we propose an optimization method for design of arithmetic circuits of arbitrary functions. For the problem of trade-off between reducing the hardware cost of the arithmetic circuit and that of the LUT, optimization of coefficients of approximated function is necessary. Furthermore, two circuit representations of the polynomial and the factorized form are used. By using the PTS, hardware model and these coefficients are optimized, and the total hardware cost is reduced. As results of simulations, the proposed method is very effective in design of arithmetic circuits of arbitrary functions.

## acknowledgment

This work is supported in part by "High-Tech Research Center Project" from the Ministry of Education, Culture, Sports, Science and Technology, Japan.

### References

[1] P.T.P.Tang, "Table-Lookup Algorithms for Elementary Functions and Their Error Analysis", Proc. 10th Symp. Computer Arithmetic, vol. 40, pp. 1, 030-1, 037, 1990.

[2] M. J. Schulte and J. E. Stine, "Approximating elementary functions with symmetric bipartite tables", IEEE Trans. on Comput., vol.48, no.9, pp.842, 1999

[3] D. T. Pham and D. Karaboga, "Intelligent optimization techniques: Genetic algorithms, tabu search, simulated annealing and neural networks", Springer, 2000.

[4] Glover, F., "Tabu Search - Part I", ORSA Journal on Computing, Vol. 1, No. 3, 1989.

[5] Glover, F., "Tabu Search - Part II", ORSA Journal on Computing, Vol. 2, No. 1, 1990.

[6] G. W. Reitwiesner, "Binary Arithmetic", in Advances in Computers, 1, Academic Press, pp. 261-265, 1960.