Abstract—In this paper, we design and implement an improved hardware-based evolutionary digital filter (EDF) version 2. The EDF is an adaptive digital filter which is controlled by adaptive algorithm based on evolutionary computation. The hardware-based EDF version 1 consists of two submodules, that is, a filtering and fitness calculation (FFC) module and a reproduction and selection (RS) module. The FFC module has high computational ability to calculate the output and the fitness value since its submodules run in parallel. However, hardware size of the FFC module is large, and many machine cycles are needed. Thus, in the hardware-based EDF version 2, we combine the two modules to reduce its hardware size and machine cycles. A synthesis result on the FPGA shows the clock frequency is 65.5MHz and the maximum sampling rate of the hardware-based EDF version 2 is 4,948.1Hz. Moreover, the hardware-based EDF version 2 is 15.7 times faster than the hardware-based EDF version 1.

I. INTRODUCTION

Several researchers have proposed adaptive algorithms for digital filtering, which are all based on the Darwinian concept of “natural selection.” These include the adaptive algorithm based on the genetic algorithm (GA) [1]–[3], the new learning adaptive algorithm [4], and Darwinian approach to adaptive notch filters [5].

The authors have already proposed evolutionary digital filters (EDFs) [6]–[8]. The EDF is an adaptive digital filter (ADF) which is controlled by adaptive algorithm based on evolutionary computation. The advantages of the EDF are summarized as follows:

1) The adaptive algorithm of the EDF is a population-based and robust optimization method, especially used to tackle high-dimensional and multi-modal search space problems. It is a non-gradient and multi-point search algorithm. Thus, it is not susceptible to local minimum problems that arise from a multiple-peak surface.

2) The EDF can adopt the various error functions as the fitness function according to application, for example, the $p$-power norm error function, the maximum error function and so on.

3) The adaptive algorithm of the EDF has a self-stabilizing feature whereby unstable poles have a tendency to migrate back into the stable region. In addition, the EDF can search the poles which are near the unit circle. Numerical examples in Refs. [6]–[8] show that the EDF has a higher convergence rate and smaller steady-state value of the square error than the LMS adaptive digital filter (LMS-ADF).

However, the EDF has the following disadvantage: the number of multiplication of the EDF is greater than that of the LMS-ADF, since the EDF consists of many inner digital filters. Thus, we implement the EDF on parallel processors.

In order to implement the EDF in parallel, we have already present a hardware implementation of the distributed EDF, that is, a hardware-based EDF version 1, which consists of the modified structure and adaptive algorithm [9]. The hardware-based EDF version 1 consists of two submodules, that is, a filtering and fitness calculation (FFC) module and a reproduction and selection (RS) module. The FFC module has high computational ability to calculate the output and the fitness value since its submodules run in parallel. However, a communication cost between modules is high, and many machine cycles are needed. Thus, in the hardware-based EDF version 2, we combine the two modules to reduce its communication cost and machine cycles.

This paper is organized as follows: Section II summarizes the overall structure and the adaptive algorithm of EDFs. Section III describes the detailed structure of the proposed hardware-based EDF version 2 and its synthesis result. Section IV gives concluding remarks.

II. EVOLUTIONARY DIGITAL FILTERING

In this section, we summarize the filter structure and the adaptive algorithm of EDFs. Figure 1 shows the block diagram of an EDF. The EDF consists of many linear/time-variant
inner digital filters $F_i$’s which correspond to individuals. Inner digital filter coefficients $W$ which correspond to the feature of individuals are controlled by the following adaptive algorithm.

A. Adaptive Algorithm of Evolutionary Digital Filters

The adaptive algorithm of the EDF is similar in concept to GA. These concepts are based on the mechanics of natural selection and genetics to emulate the evolutionary behavior of biological systems. However, the adaptive algorithm of the EDF is different from the GA in the genetic operator and the representation of strings.

In the following sections, we use the following notations:

- $P$ population of individuals,
- $N$ the number of individuals.
- The subscripts in the symbols $P$, $N$ and $W$ are denoted as follows:
  - $a$ the cloning method (the asexual reproduction),
  - $s$ the mating method (the sexual reproduction),
  - $p$ parent,
  - $c$ offspring (child).

In the EDF, the adaptive algorithm updates the inner digital filter coefficients every $T_0$ samples. Thus, the relation between the generation $t$ and the time $k$ is given by

$$k = T_0t, \ T_0t + 1, \ldots, T_0t + (T_0 - 1)$$

where $k$ denotes the time in the filtering operation and $T_0$ denotes the period of the evaluation of one generation.

1) Cloning Method: Each parent in the population $P_{ap}$, with high fitness value within the population $P(t)$, creates the offspring population $P_{ac}$ using the cloning method. In the cloning method, one parent creates $N_{ac}$ offsprings, and forms a family $P_{af,i}$ which contains itself and its offsprings, where $i = 1, 2, \ldots, N_{ap}$. $N_{ap}$ is the number of parents which use the cloning method. We assume that the proposed cloning method corresponds to transcribing the coefficient vector $W_{ap,i}$ as the parent feature into coefficient vectors as the offspring feature $W_{ac,i,j}$, where $i = 1, 2, \ldots, N_{ap}$ and $j = 1, 2, \ldots, N_{ac}$.

Thus, the proposed cloning method updates the inner digital filter coefficients as individual feature according to

$$W_{ac,i,j} = W_{ap,i} + r \cdot n_{i,j}$$

where the scalar $r$ denotes the cloning fluctuation, and $n_{i,j}$ is a Gaussian random variable vector with zero mean and unit variance.

In this algorithm, the cloning method corresponds to the local search. Therefore, this method is provided with the following strategy to select the candidate population for the next generation. In this method, one individual, of which fitness is maximum in each family $P_{af,i}$, is selected. These individuals form the candidate population $P_a$ for the next generation. The population $P_a$ of the best individuals is selected among each family $P_{af,i}$, that is, the coefficient vector of the inner filter with the highest fitness is selected among the $(N_{ac} + 1)$ coefficient vectors. These coefficients are scattered on the narrow area. Thus, this operation corresponds to the local search.

2) Mating Method: If parents with low fitness value in population create the offsprings using the above cloning method, these offsprings may have low fitness value and can not be selected as candidates for the next generation. Therefore, parents in the population $P_{sp}$, with low fitness value within the population $P(t)$, are created the offspring population $P_{sc}$ using the mating method. $N_{sp}/2$ pairs among the $N_{sp}$ parents are randomly selected for mating. In the mating method, each pair of parents creates one offspring, and they form a family $P_{sf,m}$ which contains themselves and their offspring, where $m = 1, 2, \ldots, N_{sp}/2$. $N_{sp}$ parents are randomly selected for mating. Therefore, this operation corresponds to the global search and keeps various features of individuals.

The subscripts in the symbols $P$, $N$ and $W$ are denoted as follows:

- $a$ the cloning method (the asexual reproduction),
- $s$ the mating method (the sexual reproduction),
- $p$ parent,
- $c$ offspring (child).

Thus, the proposed cloning method updates the inner digital filter coefficients as individual feature according to

$$W_{sc,m} = \frac{1}{2}(W_{sp,k(m)} + W_{sp,l(m)}) + s \cdot n_m$$

where $k(m)$ and $l(m)$ are selected in $\{1, 2, \ldots, N_{sp}\}$ without duplicating, and $m = 1, 2, \ldots, N_{sp}/2$. The scalar $s$ denotes the mating fluctuation, and $n_m$ is a Gaussian random variable vector with zero mean and unit variance.

In this algorithm, the mating method corresponds to the global search and keeps various features of individuals. Therefore, this method is provided with the following strategy to select the candidate population for the next generation. In this method, one parent with higher fitness value in each family $P_{sf,m}$ is selected and the other parent dies out. In order to keep various features of individuals, the offspring in each family $P_{sf,m}$ is always selected regardless of their fitness values.

B. Computational Complexity

The EDF requires $(A - N_{ap} - \frac{1}{2}N_{sp})(N + M + 1)/T_0$ multiplications per iteration for the adaptive process, where $A$ is the total number of the evaluated individuals, that is, $A = N_{ap}(N_{ac} + 1) + \frac{3}{2}N_{sp}$. In the adaptive algorithm of the EDF, the inner digital filter coefficients are updated every $T_0$.
samples. Table I shows that the number of multiplications of the EDF is larger than that of the LMS-ADF.

III. HARDWARE-BASED EDF VERSION 2

Figure 2 shows the block diagram of the hardware-based EDF version 1. The EDF module consists of two submodules, that is, a filtering and fitness calculation (FFC) module and a reproduction and selection (RS) module.

The FFC module has high computational requirement, since the FFC module performs filtering and fitness calculation of a large number of individuals. Thus, the FFC module has single filtering modules (SFMs) which are submodules and perform filtering and fitness calculation per individual. The FFC module has the high computational ability to calculate the output and the fitness value since the SFMs run in parallel.

The RS module consists of the following modules: a single reproduction and selection (SRS) module which perform a reproduction and selection operation every individual, and an SRS control module. The RS module, first, repeats the following steps in parallel until fitness values of all individuals in population are evaluated.

Step 1. Reproduce an individual according to fitness value in population.
Step 2. Send an information of the individual to the FFC module.
Step 3. Receive a fitness value of the individual from the FFC module.

Second, the individual which has the maximum fitness value in population is selected. Finally, the output for the EDF is selected from the common memory.

This structure can perform parallel processing efficiently, since these modules work in parallel. Moreover, using this structure, it is easy to design these modules and write HDL code for them. However, a communication cost between modules is high, and many machine cycles are needed.

The output of the EDF is the output of an inner filter for which fitness value is maximum. Therefore, the output of the EDF is selected after all fitness values of inner filters is evaluated. Thus, the EDF module has a common memory to keep output signals of all inner digital filters throughout $T_0$ samples every iteration as shown in Figure 2.

In order to improve the hardware-based EDF version 1, we design and implement a hardware-based EDF version 2 to reduce the hardware cost and increase the sampling rate.

A. Hardware Structure of Evolutionary Digital Filters

Figure 3 shows the block diagram of the hardware-based EDF version 2. The EDF module consists of two submodules, that is, a filtering, fitness calculation and updating coefficients module, and a sorting module. The former consists of the FFC module and a part of the RS module in the hardware-based EDF version 1. The latter is the sorting submodule of the RS module in the hardware-based EDF version 1.

Table II shows specifications of the hardware-based EDF version 2. Format of signals and coefficients on the hardware-based EDF is “Q14,” that is, 16-bit fixed-point format with an integer part in the high-order 2bits and a fractional part in the low-order 14bits in consideration of the range of the coefficients.

B. Filtering, Fitness Calculation and Updating Coefficients Module

In the hardware-based EDF version 1, the assembly language is used to control the FFC module, since it is easy to change the filter order, the filter structure and the fitness function. However, it requires a large number of machine cycles.

Therefore, to reduce the number of machine cycles and the hardware cost, we restrict the fitness function to the mean square error, and the filter structure to the direct-form II IIR filter. In addition, the filtering and fitness calculation module in the hardware-based EDF version 2 is controlled using a state machine.

Table III shows machine cycles per individual for processing one sample of the hardware-based EDF version 1 and 2. It is
shown that the numbers of machine cycles for all submodules are reduced.

In the hardware-based EDF version 1, the FFC module and the updating coefficient submodule of the RS module have a similar structure, that is, they have a MAC (multiplier and accumulator) and a similar data path. In addition, in order to control the FFC, its controller has 180 states, since the machine language for the FFC consists of the 45 instruction, and its controller has four stages (Fetch, Decode, Read and Execute). Thus, in the hardware-based EDF version 2, we combine the two modules to reduce the hardware cost and the communication cost. The new module has a MAC and a data path, and is controlled using a state machine which has only 24 states. In addition, the program and data memory, which is $512 \times 16$ bits, for the FFC is reduced.

Table V shows the synthesis results of the filtering, fitness calculation and updating coefficients modules. Table IV shows an FPGA and a tool to implement the hardware-based EDF version 1 and 2. It is shown that the hardware size of the version 2 is 1/3.6 times that of the version 1, and the clock frequency of the version 2 is 1.8 times faster than that of the version 1.

### C. FPGA Implementation

Table VI shows Synthesis results of the hardware-based EDF version 1 and 2. Table IV shows an FPGA and a tool to implement the hardware-based EDF version 1 and 2. In addition, Table III shows machine cycles per individual for processing one sample of the hardware-based EDF version 1 and 2.

These results show that the hardware size of the version 2 is 1/3.3 times that of the version 1, and the clock frequency of the version 2 is 2.3 times faster than that of the version 1.

The maximum sampling rate of the version 2 is 4,948.1Hz, since the number of cycles of the version 2 is 1/6.4 times that of the version 1. Therefore, the maximum sampling rate of the version 2 is 15.7 times faster than that of the version 1. Moreover, the maximum sampling rate of the hardware-based EDF version 2 is 2.9 times faster than that of the software-based EDF.

### IV. Concluding Remarks

In this paper, the hardware-based EDF version 2 which is the improved version of the hardware-based EDF version 1 [9] has been designed and implemented. Its maximum sampling rate is 4,948.1Hz.

In the future, the distributed EDF presented in Ref. [10] is applied to the hardware-based EDF version 2 to improve the maximum sampling rate.

### REFERENCES


