

A Genetic Algorithm for the Optimisation of a Reconfigurable Pipelined FFT Processor

Nasri Sulaiman and Tughrul Arslan
Department of Electronics and Electrical Engineering
The University of Edinburgh
Scotland

N.Sulaiman@ed.ac.uk, Tughrul.Arslan@ee.ed.ac.uk.

Abstract

This paper describes the optimisation of the word length in a 16-point radix-4 reconfigurable pipelined Fast Fourier Transform (FFT) based receiver device. Two forms of optimisation; input data optimisation and FFT coefficients optimisation are investigated in this paper. The word length for input data and FFT coefficients are initially set to 16-bits. A Genetic Algorithm (GA) is then used to find the optimal word length for the input data and FFT coefficients while satisfying functionality constraints. The GA is able to determine an optimised word length down to 10 bits for input data and 8 bits for the FFT coefficients.

1. Introduction

1.1. Evolvable Hardware (EHW)

Evolvable Hardware (EHW) is a type of hardware platform which can reconfigure its structure dynamically (on-line) and autonomously, according to changes in task requirements or in the environments in which it is embedded. The interest in research on EHW is growing rapidly after it was initiated independently in Japan and in Switzerland around 1992 [1].

One of the areas which EHW has been currently applied to is digital signal processing (DSP). One example of the applications in this area is the design of finite impulse response (FIR) filters. Some examples of the designs of the FIR filters are the work done by Suckley, Dexiang et. al [2], Tufte et. al [3] and Thomson et. al [4].

Another example of the applications of EHW in DSP is the design of infinite impulse response (IIR) filters. Some examples of the designs are the work done by Wilson et. al, and Arslan et. al [2].

Fast Fourier Transform (FFT) is another potential example of the application of EHW since it is widely used in various applications such as telecommunications, radars, speech/image processing, medical electronics, and seismic processing, etc. Specifically, FFT processors are

key building blocks in multi-user mobile telecommunication transceivers, where optimum hardware performance in terms of power, area, and speed is of prime importance. An invaluable resource optimisation technique would be that of dynamically adapting the receiver architecture subject to environmental conditions, such as delay spread etc, at a given instant in time.

The word length will affect both the performance and the hardware complexity of the FFT [9]. Therefore it is desirable to design a reconfigurable FFT in term of the word length of the input data and FFT coefficients. This paper describes a work on optimising the word length of the input data and FFT coefficients using genetic algorithm (GA).

1.2. Pipelined Fast Fourier Transform

Fast Fourier Transform (FFT) plays an important role in the design and implementation of discrete-time signal processing algorithms and systems. In recent years, there has been a tremendous growth in the design of the high-performance FFT processors due to the emerging applications in the modern digital communication systems and television terrestrial broadcasting systems. These systems require real-time FFT performance to support certain critical operations such as orthogonal frequency division multiplexing (OFDM) and motion compensation [10].

Implementation of FFT on different architectures, for fast and efficient computational schemes has attracted many researchers since mid 1970s [5]. Malladi and et. al., [5] proposed VLSI realisation of FFT algorithm which, (i) have pipelining and/or parallelism, (ii) be regular and modular, (iii) use small number of basic cells repeatedly and (iv) the basic processing element should be suitable for integration with current technology.

Pipelined FFT processor is a class of real-time FFT architectures which continuously process the input data which usually arrives in the word sequential format for the reason of the transmission economy. It utilises concurrent processing of different stages to achieve high throughput.

The advantages with this architecture are high data throughput, relatively small area and a relatively simple control unit.

2. FFT Design

The N-points Discrete Fourier Transform (DFT), $X(k)$ of N-points sequence $x(n)$ is defined by [6]

$$X(k) = \sum_{n=0}^{N-1} x(n)W_N^{nk}, k \in [0, N-1] \quad (1)$$

where $W_N = e^{-j\frac{2\pi}{N}}$ (2)

The first equation shows that the complexity of a direct realisation of this algorithm is $O(N^2)$. In order to reduce the computation complexity of DFT, an FFT algorithm uses a divide and conquer approach.

The first design issue on the algorithmic level is to select the radix of the FFT. It will determine the time taken for computation and area [7]. In this work, radix-4 is chosen because it has a computational advantage over radix-2. Radix-4 butterfly does the work of four radix-2 butterflies using three multipliers instead of four multipliers in four radix-2 butterflies [8].

The next design issue is to choose which radix-4 pipelined FFT architectures. There are various architectures for radix-4 pipelined FFT, for examples, radix-4 single-path delay feedback (R4SDF), radix-4 multi-path delay commutator (R4MDC) and radix-4 single-path delay commutator (R4SDC) [5]. In this work, R4SDC architecture is chosen because it has been used recently in building the largest ever single chip pipelined FFT processor for HDTV application [10]. Moreover, the utilisation factor in this architecture is 100% and 75% for the butterfly and multiplier respectively for sequential input data [8]. This results in tremendous saving in hardware and therefore power consumption for real-time applications [8].

The algorithm for this FFT architecture is derived from (1). Let N be a composite number of v integers so that $N = r_1 r_2 \dots r_v$ and define

$$N_t = N / r_1 r_2 \dots r_t \quad 1 \leq t \leq v-1$$

Where t is the stage number of the decomposed DFT and r_t its radix. The pipelined FFT processor is obtained by decomposition an N-point DFT into v stages. The final stage is defined in [8] as follows.

$$X(r_1 r_2 \dots r_{v-1} m_v + \dots + m_1) = \sum_{q_{v-1}=0}^{r_v-1} x_{v-1}(q_{v-1}, m_{v-1}) W_{r_v}^{q_{v-1} m_v} \quad (3)$$

Whereas intermediate stages (t) are given by following recursive equation.

$$x_t(q_t, m_t) = W_{N_{t-1}}^{q_t m_t} \sum_{p=0}^{r_t-1} x_{t-1}(N_{t-1} p + q_t, m_{t-1}) W_{r_t}^{p m_t} \quad (4)$$

where

$$2 \leq t \leq v-1, 0 \leq m_i \leq r_i-1, 0 \leq q_i \leq N_i-1, \text{ and } 2 \leq i \leq v$$

For $r_1 = 4$, the corresponding equations are as follows.

$$X(4m_2 + m_1) = \sum_{q_1=0}^3 x_1(q_1, m_1) W_4^{q_1 m_2} \quad (5)$$

$$x_1(q_1, m_1) = W_{16}^{q_1 m_1} \sum_{p=0}^3 x_1(4p + q_1) W_4^{p m_1} \quad (6)$$

where $0 \leq m_1, m_2 \leq 3$

Figure 1 displays the flow graph of a 16-point FFT based on the above formulation. In Figure 1, each open circle represents the summation while the dots define the stage boundaries. The number inside the open circle is the value of m_1 (stage 1) or m_2 (stage 2). The number outside the open circle is the FFT coefficient applied [8].

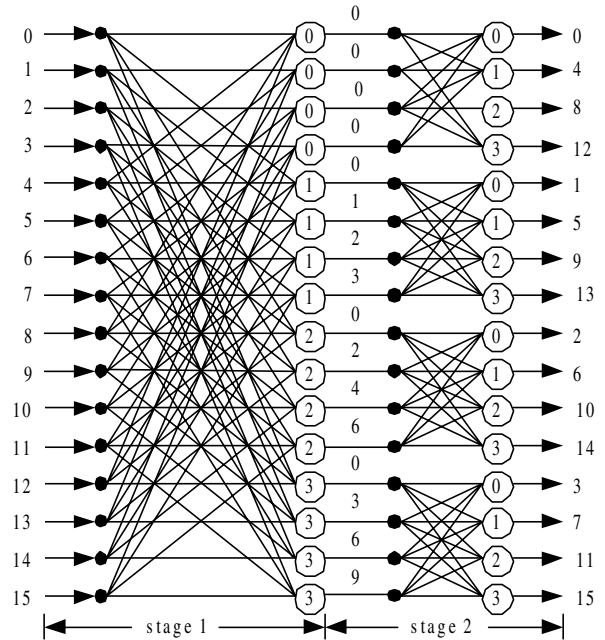


Figure 1: Signal flow graph of a radix-4 16-point FFT

A 16-points R4SDC pipelined FFT processor which is used in this work is shown in Figure 2.

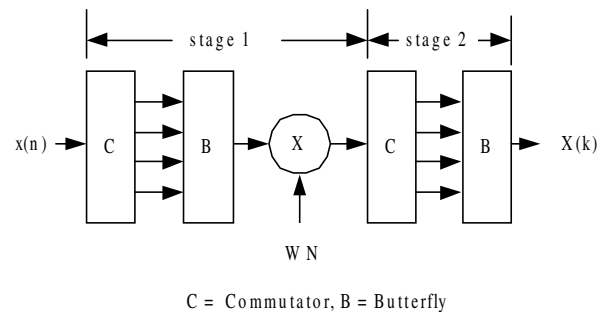


Figure 2: Block diagram of a 16-point R4SDC pipelined FFT

3. Genetic Algorithm (GA)

3.1 Representation Method

In term of the GA gene, 16 input data and 16 FFT coefficients in complex form are represented as a string comprising of sequences of 32 W bit binary numbers where W is the FFT word length. The representation of the above 16-point FFT is shown in Figure 3.

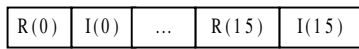


Figure 3 : Gene representation for input sequences and FFT coefficients

3.2 The Initial Population

In our implementation, the initial population for the input data is comprised of randomly selected 16 complex numbers of 16 bits. The initial population for the FFT coefficients is obtained in this way. First, the coefficients are calculated using equation (2). These coefficients in the floating point form are then converted into the 16 bits binary numbers format. Next, the above 16 bits numbers are rounded to the nearest finite word length number and then transformed to a finite range value using a random process. In the random process, uniformly selected numbers in the range [-5,+5] are arbitrarily chosen as the initial values. This range of numbers is then changed after evaluating the solution. This step is repeated several times until a satisfactory condition to provide the solution is achieved. From the experiments, uniformly selected numbers in the range [-4,+4] and [-6,+6] were found to be satisfactory to breed promising solutions through successive generations for the input data and FFT coefficients respectively. A population of 50 is arbitrarily selected as the starting point for the initial population. This number is varied a few times after evaluating the solution until a satisfactory level of genetic diversity is found. From the experiments, the initial populations with the size of 50 and 100 were found to provide sufficient genetic diversity for the input data and FFT coefficients optimisations respectively.

3.3 Genetic Operators

The operation of GA begins with evaluating the fitness of the initial population. This is followed by the operation of selecting parent chromosomes using roulette wheel selection. This consists of a weighted probability scheme with selection according to the fitness of each chromosome, such that FFT chromosomes have greater likelihood of being selected as parents [2]. Crossovers between selected parents are then performed at a rate of 90%. The next operation is mutation where a member of

the current population is randomly selected and replaced with new value given by a random process. Mutation is carried out at a rate of 10%.

3.4. Fitness Function

Fitness is evaluated by comparing the spectrum power of each parent with the specification. The specification is obtained by calculating the spectrum power for the 16 bits input data and 16 bits FFT coefficients. For each evaluation, a violation is checked for where the magnitude of spectrum power of each parent is lower than the specification. If a violation is detected then an error value is calculated (as the difference between the parents and specification) and then squared. This is repeated for all the outputs. Finally, a total error is evaluated as defined by the following:

$$\Delta = \sum_{i=0}^{15} W_i (Xps_p - Xps_c)^2 \quad (7)$$

Where,

Xps_p = spectrum power of the parent

Xps_c = spectrum power of the specification

W_i = error weighting function

In evaluating the total error, the initial value for the error weighting function is arbitrarily set to 1.000. This figure is then modified after evaluating the solution. This step is repeated several times until a satisfactory condition to find the first best coefficients is obtained. From the experiments, uniform error weightings of 0.001 and 0.01 were found to give a promising condition to achieve the first best coefficients for the input data and FFT coefficients optimisations respectively.

4. Results

4.1. Input Data Optimisation

The objective of this experiment was to find the optimised word length for the input data. Initially, the word length for the input data and the FFT coefficients were set at 16 bits. The word length of the input data was reduced from 16 bits to 14 bits. GA was used to search for the best coefficients for the input data. This experiment was repeated with the input data which has the word length of 12 bits. Figure 4 shows the GA search results for both 14 bits and 12 bits word length. From, the figure, the first best coefficients for the 12 bits and 14 bits were obtained in the third and first generation respectively. This shows that the optimisation to 14 bits and 12 bits was a relatively easy task since the solutions can be obtained in the very early generation.

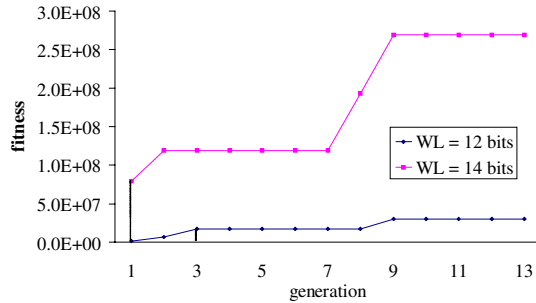


Figure 4: GA search results for 12 bits and 14 bits word length of the input data

The word length of the input data was reduced further from 12 bits to 10 bits to increase the level of the difficulty in the optimisation search. The GA again was used to look for the best coefficients for this word length. Figure 5 displays the GA search results for the 10 bits word length. As shown in Figure 5, GA found the first best coefficients in the 20 generation. This GA program was running at the Sun Blade -100 workstation and it took approximately 0.490s to obtain the complete solution. The search also found another 2 best solutions which were in the 23 and 24 generations respectively.

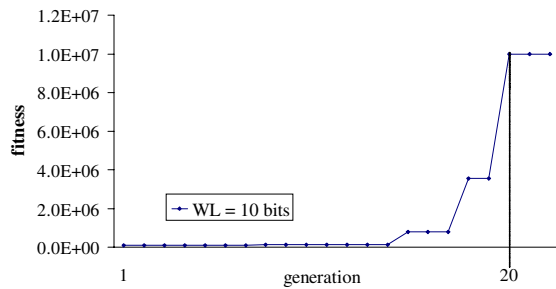


Figure 5 : GA search results for 10 bits word length of the input data

Spectrum power was calculated for the best coefficients obtained from Figure 5. This spectrum power was then compared with the specification as shown in Figure 6.

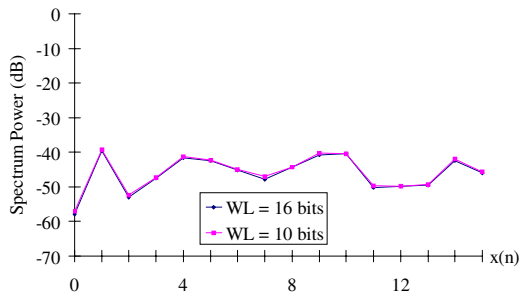


Figure 6 : Spectrum power for 16 bits and 10 bits word length of the input data

As shown from Figure 6, the spectrum power for the 10 bits word length is close to the 16 bits word length. In addition, Figure 7 displays the deviation of the spectrum power for the 10 bits word length from the 16 bits word length.

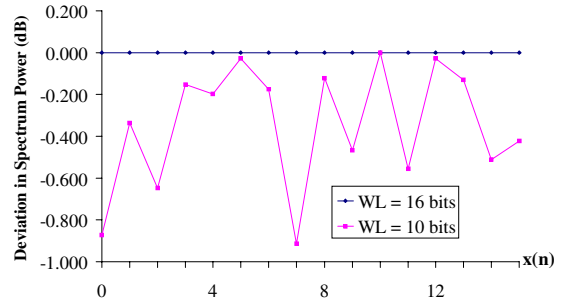


Figure 7 : Deviation of the spectrum power for the 10 bits word length

The largest deviation is 0.9 dB which is a very good indication that the spectrum power for the 10 bits word length is very close to the 16 bits word length

4.2. FFT Coefficients Optimisation

The purpose of this experiment was to find the optimised word length for the FFT coefficients. The word length of the FFT coefficient was reduced from 16 bits to 8 bits and the word length for the input data was fixed at 16 bits. Then the GA was used to search for the best coefficients for the 8 bits word length. Figure 8 displays the GA search results for the 8 bits word length. In this search, the first best coefficients were found in the 72 generation. The GA search took approximately 1.110s to obtain the complete solution.

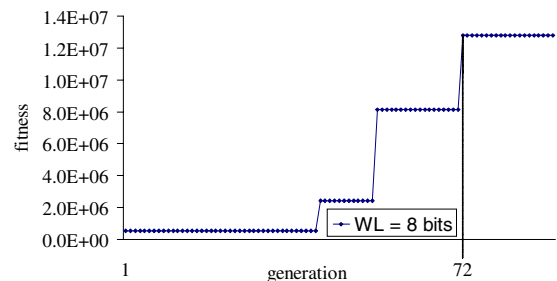


Figure 8 : GA search results for 8 bits word length of the FFT coefficients

Table 1 shows the output of the FFT obtained by GA for the coefficients in the 72 generation compared to the output given by MATLAB. The outputs obtained from GA are close to the ones provided by the MATLAB.

	GA		MATLAB	
	REAL	IMAG.	REAL	IMAG.
X(0)	0.076998	-0.045961	0.076998	-0.045961
X(1)	0.160162	-0.717954	0.159470	-0.717750
X(2)	-0.091856	0.138043	-0.073809	0.140156
X(3)	0.103776	-0.299294	0.094171	-0.282842
X(4)	0.573992	-0.143040	0.573992	-0.143040
X(5)	-0.516214	-0.145189	-0.514754	-0.149358
X(6)	0.131292	-0.381862	0.133953	-0.369916
X(7)	-0.171231	0.249916	-0.161063	0.234050
X(8)	0.424970	-0.044008	0.424970	-0.044008
X(9)	0.346611	-0.547399	0.360220	-0.538791
X(10)	0.649857	0.220000	0.631809	0.217887
X(11)	0.103910	-0.213419	0.094045	-0.197672
X(12)	-0.224006	0.029023	-0.224006	0.029023
X(13)	0.245531	0.062601	0.231155	0.057956
X(14)	0.242743	0.491973	0.240082	0.480027
X(15)	-0.248376	0.274760	-0.239073	0.258428

Table 1 : Outputs of FFT obtained by GA and MATLAB

Figure 9 shows the comparison of the spectrum power for best coefficients obtained for the 8 bits word length with the 16 bits word length. The spectrum powers for the 8 bits word length is close to the 16 bits word lengths.

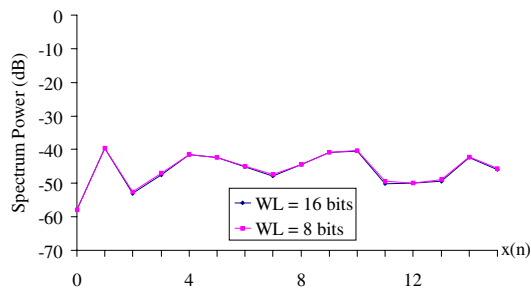


Figure 9 : Spectrum power for 16 bits and 8 bits word length of the FFT coefficients

The deviations in the spectrum power of the 8 bits word length from the 16 bits word length are shown in Figure 10. The maximum deviation was 0.7 dB which is also a good indication that the spectrum power for the 8 bits word length is very close to the 16 bits word length.

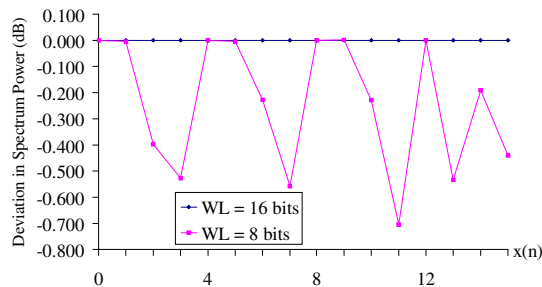


Figure 10 : Deviation of the spectrum power for the 8 bits word length

5. Conclusion

In this paper we have described the word length optimisation of the input data and FFT coefficients using a GA. Results show that our GA is able to find optimised coefficients using 8 bits with the input data requiring only 10 bits. Currently the GA is being targeted for an on-line adaptation of a 16-bit reconfigurable FFT processor which is a part of a MCDMA receiver.

6. References

- [1] I. Kajitani, T. Hoshino, M. Iwata, and T. Higuchi, "Variable length chromosome GA for Evolvable Hardware", in *Proceedings of IEEE International Conference on Evolutionary Computation*, pp. 443–447, May 1996.
- [2] T. Arslan and D. H. Horrocks, "A Genetic Algorithm for the Design of Finite Word Length Arbitrary Response Cascaded IIR Digital Filters", in *First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*, pp. 276–281, 1995.
- [3] G. Tufte, and P. Haddow "Evolving an Adaptive Digital Filter", in *Proceeding of the Second NASA/DoD Workshop on Evolvable Hardware*, pp. 143–150, July 2000.
- [4] R. Thomson and T. Arslan, "Evolvable Hardware for the Generation of Sequential Filter Circuits", in *2002 NASA/DoD Conference on Evolvable Hardware*, pp. 17–25, July 2002.
- [5] S. R. Malladi, S. R. Myneni, P. V. Pothana and M. A. Bayoumi, "A High Speed Pipelined FFT Processor", in *International Conference on Acoustics, Speech, and Signal Processing*, pp. 1609–1612, April 1991.
- [6] J. G. Proakis and D. Manolakis, *Digital signals processing: Principles, algorithms and applications*, Macmillan publishing company, 1992.
- [7] S. He and M. Torkelson, "A New Approach to Pipeline FFT Processor", in *Proceedings of IPSP '96., the 10th International Parallel Processing Symposium Conference*, pp. 766–770, May 1996.
- [8] M. Hasan, *Low Power Techniques and Architectures for Multicarrier Wireless Receiver*. PhD thesis, University of Edinburgh, 2003.
- [9] S. Johanson, S. He, and P. Nilsson, "Wordlength Optimization of a Pipelined FFT Processor", in *42nd Midwest Symposium on Circuits and Systems*, pp. 501–503, August 1999.
- [10] Yun-Nan Chang, and K. K. Parhi, "Efficient FFT Implementation Using Digit-Serial Arithmetic", in *1999 IEEE Workshop on Signal Processing Systems*, pp. 645–653, October 1999.