ISSN 2278-3091
Volume 8, No.1.6, 2019
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse5681.62019.pdf
https://doi.org/10.30534/ijatcse/2019/5681.62019

Optimization of Fast Fourier Transform processor using Genetic Algorithm on Raspberry Pi

Firas Faisal Ghazi, Nasri Sulaiman
Department of Electrical and Electronic Engineering Faculty of Engineering
Universiti Putra Malaysia 43400 Serdang, Selangor

ABSTRACT

In many areas of engineering and science are widely using FFT processors in most of their applications, thus, the modern science requires continuously new optimizations which includes the FFT processor to have a lower power consumption and a smaller size. This paper revolves around the 16-point Radix-4 Single Path Delay Feedback (R4SDF) for optimizing the pipelined Fast Fourier Transform (FFT) processor, which can be done by using both Single Objective Genetic Algorithm (SOGA) and Multi-Objective Genetic Algorithm (MOGA), Signal to Noise Ratio (SNR) value which depends on the word length of the FFT processor, higher word length value of the FFT processor will result in a higher value for the SNR, Thus, this research aims to reduce the power consumption of the FFT processor by lowering the word length of Twiddle Factor for the FFT by using a SOGA to find the optimum results for SNR values while MOGA is set to find the optimum values for both SNR and SA while lowering the Word Length. Over the years the Genetic Algorithms (GA) proved to be one of the best methods for optimization. In this research the GA is for reducing the word length by optimizing its coefficients. The required amount of value for the SNR is to be more than 63 dB and less than 192 for SA. The proposed work was done successfully in optimizing the FFT by using SOGA to lower the word length until 12 bits and obtaining a SNR value of 66.452 which resulted in an improvement of 5.47% for SNR while MOGA was achieve a value of 65.033dB which provides (3.22%) improvement for the SNR and SA value of 140 which made reduction by (27%) for the SA.

Key words : FFT processor, Genetic Algorithm, Signal to Noise Ratio, Switching Activity, SOGA, MOGA.

1. INTRODUCTION

The FFT processor is the most critical block in digital communication systems (DSP) such as MC-CDMA receiver. Power consumption has been a major concern in the digital communication systems. Power consumption depends on the Word Length of the Twiddle Factor for the FFT processor [1], Twiddle Factors are the results of breaking down the DFT algorithm of N size into smaller sizes of N1,N2 with O(N) multiplications [2]. One way to reduce the power consumption in the FFT processor is to reduce the Word Length for the Twiddle Factor, to lower the Word Length, it must meet the target objectives for both of Signal to Noise Ratio (SNR) which is considered as the accuracy factor and Switching Activity (SA) which is considered as the power consumption factor [3]. Most of the researchers who optimized the FFT processor used Genetic Algorithm, the Genetic Algorithm (GA) can be tasked to find the best solutions for the most complex problems, it can be tasked as a Single-Objective Genetic Algorithm (SOGA) or a Multi-Objective Genetic Algorithm[4]. The algorithm normally can be implemented on a device such as Raspberry Pi, although a lot of researches focused on the optimization of FTT for the power consumption none tried to optimize FFT on the Raspberry Pi. Raspberry Pi is a single board micro-computer which can be programed and tasked to do all kinds of research, For the proposed work the Raspberry Pi was able to give us the same results as a fully functional computer, while most of the previous researchers worked on fully functional computers to get their results that is why the Raspberry Pi can be considered as a cost efficient device.

2.1 Pipelined FFT

A pipelined FFT architecture is so important because it does not require a lot of butterfly unit and a lot of memory sizes and it provides a moderate hardware complexity as well as a regularity of data path and a better performance speed. The pipelined FFT structure can be divided into two main types which are : the Delay Commutator (DC) and the Delay Feedback (DF) Depending on the steam paths of the input data they as well can be divided into two categories : Multiple Path or Single path architectures [5]

2.2 Radix-4 pipelined FFT processor

The Radix-4 is designed to simplify the Radix-2 stages by half of the stages required, for example to calculate an input of 16 point FFT, in order to do that, the Radix-2 will take four number of stages while the Radix-4 will take 2 stages as shown in Figure 1[2].

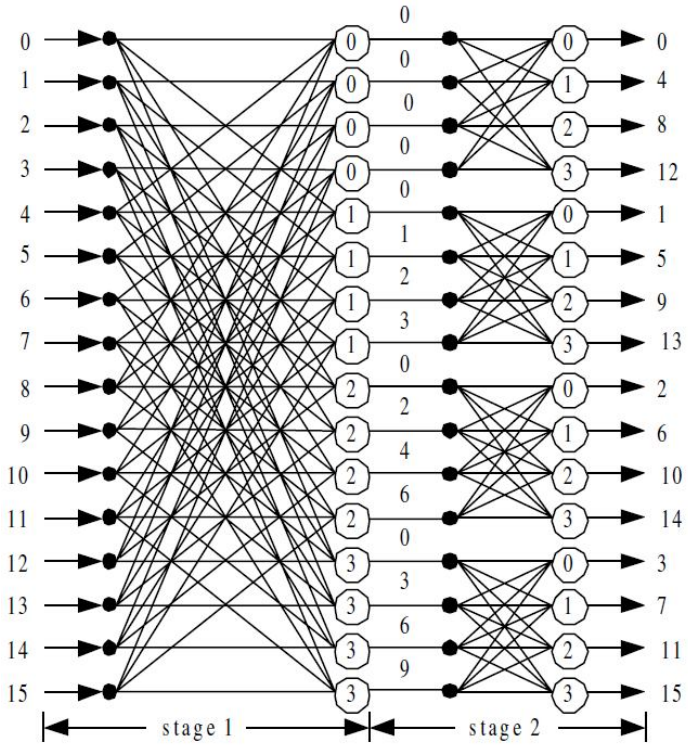

Figure 1: Structure of 16 point Radix-4 DIF FFT

3.1 Genetic Algorithm implementation

Summarizing the whole implementation of the proposed work in Figure 2 in the steps below

1- The Genetic Algorithm will be downloaded and executed on the Raspberry Pi.
2- The Genetic Algorithm will use the initial value of the Twiddle Factor to obtain the new Twiddle Factor outputs.
3- Twiddle Factor outputs will be the feedback to Genetic Algorithm to obtain the values for the SNR and SA in which it will be used to optimized the Word Length of the Twiddle Factor.

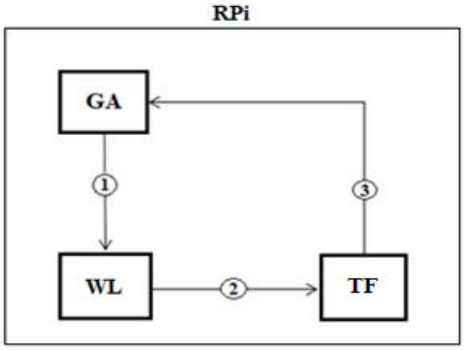

Figure 2 : General block diagram for proposed work

3.2 Genetic Algorithm

GA works by taking a set of data (chromosomes) as the targeted data to be evolved, it selects (parents) from the chromosome then performs crossover and mutation processes[6][7], the crossover and mutation rates can vary due to the nature of the selected work, by adjusting these two factors the solution can be obtained, which is referred to as the (child) from the parents, the fitness of the child is compared to the required solution, if the child matches it or not, the child with the low fitness will most likely be discarded while the child with the higher fitness will be matched with another child with a high fitness which they will be counted as the parents and the cycle process of making the solution (child) continues until it comes up with the best solution. Figure 3 shows the representation of chromosomes after crossover.

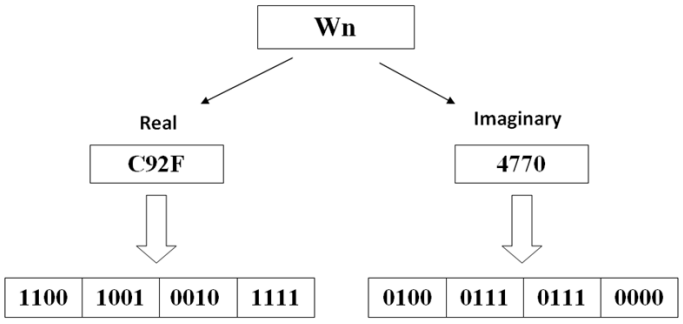

Figure 3: Chromosome representation in Genetic Algorithm

When representing the data string the word length of the FFT coefficients (which is 32 bits) is divided to two values
1- Real Data (16 bits Real).
2- Imaginary Data (16 bits Imaginary).

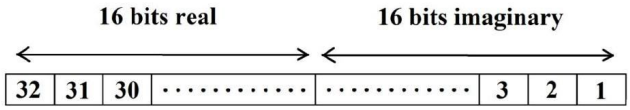

Figure 4 : Real and Imaginary Data Represented in 32 bits Word Length

When the Genetic Representation is done, the randomization process is applied to create an initial population, if a coefficient is subjected to a total randomization in the randomization process, it will lead the optimization to take a long time and makes the search even more difficult than the expected procedure [8][9]. Thus, to avoid such circumstance, the semi-randomization is applied by using it on the least significant bits (LSB) as shown in the Figure 5

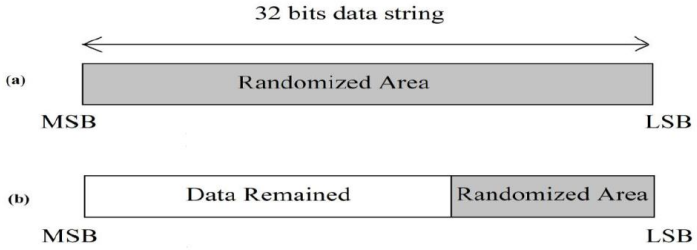

Figure 5 (a) : Full Randomization **(b**) Semi-Randomization

a basic GA structure is shown in Figure 6

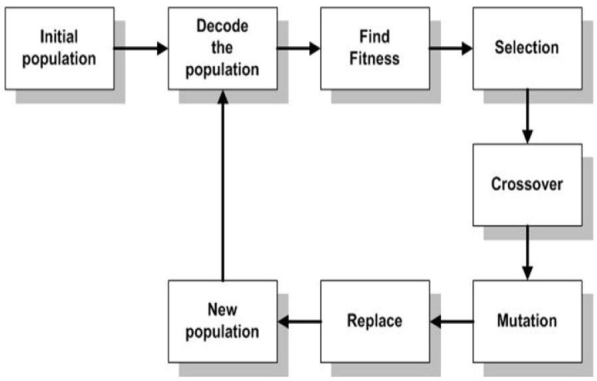

Figure 6: block diagram of a simple genetic algorithm

3.3 Initial Population

The GA requires an initial population in order to start finding solution, and In this research the initial population is a set of Twiddle Factors [10], therefore we selected(from the GA output) 16 coefficients of 16 bits each as an input data and it is considered as the starting point for the GA compilation, first we calculate the Twiddle Factor for each coefficient, next, we transform those coefficients from their hexa-decimal forms to their binary forms, then, we use round these coefficients to the closest finite word length number and use a random process to convert them to a values of finite range[11]. The initial values for random process are randomly selected between the range of [-5 to +5], and this range is also subjected for change after getting the solutions. To get the required solution this step is repeated several times.

3.4 Crossover

In each population, there are 16 coefficients in the 16 point R4SDF FFT. The crossover point is selected in a set of coefficients as shown in Figure 7, by doing the crossover point, it can exchange the information between any two populations by randomizing the numbers of coefficients form for the set of 16 coefficients. for example, a coefficient number 7 is chosen as the crossover point, which leads to the coefficient values for the first solution (from 7 to 16) will exchange with the other solution. The quality of the optimization relies on the exchange between these solutions.

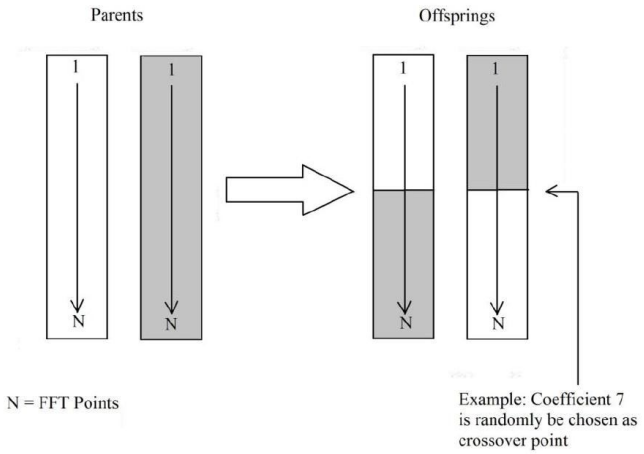

Figure 7 : Randomly Chosen Crossover point for set of Coefficient

According to GA searches [3] [4], the estimated value for the GA search crossover rate range between 0.6 to 1.0. In this research the crossover rate is set to a value of 0.9 for the crossover process and a single point crossover is applied as well.

3.5 Mutation

The mutation is the element in GA that adds flexibility to generate new data. As For the mutation rate , the higher mutation rate will cause a great amount of undesired quality of chromosome, so it is better to keep the mutation rate to a low value of 0.1. As for the population size used in the GA search, it is set to a value of 50, the normal value is set to be in range between 20 to 100 according to [12] [13]. A larger population will cause a faster optimization, but it demands an implementation of even larger hardware size.

3.6 Signals to Noise Ratio

To get the SNR value by using equation

$$SNR = 10log_{10}\left[\frac{\sum(R_{16})^2+\sum(I_{16})^2}{\sum(R_{16}-R_{wl})^2+\sum(I_{16}-I_{wl})^2}\right]$$

In the equation, before the optimization for the FFT, R_{16} and I_{16} are used as real and imaginary parameters for the FFT output, after the optimization of the FFT output R_{wl} and I_{wl} are considered as the real and imaginary parameters of the FFT output. In order to get the SNR, it starts with calculating the difference between the optimized value and non-optimized value for the GA as shown $\sum(R_{16}-R_{wl})^2+\sum(I_{16}-I_{wl})^2$. Then calculate the non-optimized value $\sum(R_{16})^2+\sum(I_{16})^2$ and divided them as shown in equation. The proposed equation in [14] is based on the general SNR equation used by [15].

3.7 Switching Activity

To calculate the SA, we use the Hamming Distance calculation. As for the calculation of the fitness, it can verified by calculating the difference between the total switching activity value and the switching activity value, for example, if

a 32 bit word length and Twiddle Factor of 16, the calculation for the total SA will be 32 multiplied by 16 which gives the value of 512. The value of the non-switching is referred as (Total_SA-SA).

$$SA\_fitness = \frac{Total\_SA - SA}{Total\_SA}$$

if SA is high, it will lead to a low fitness, any individual in the GA which has a low SA will give a high SA_fitness.

3.8 Single Objective GA Results

The purpose of the SOGA is to find the optimum value for SNR which will optimize the effectiveness for the Twiddle Factor for the FFT. In order to do that the SOGA will look for the best objective solutions for the SNR which must meet the goal of getting higher than 63dB, any value that is higher than 63 dB is considered as an acceptable solution, the default word length for the FFT is 16 bits, the SOGA will start searching from 15 and below to get values that meet our SNR's goal, we will decreasing the bits by 1 bit each time and check how many acceptable solutions that can be found, we will keep doing that until it won't be able to find a value more than 63 dB.

4.1.1 SOGA results for word length of 15 bits

Table 1 shows the Twiddle Factor results that were used to find the highest SNR value which is shown in Figure 8. a 15 bits of Word Length means XXXX XXXX XXXX XXXO of digits, in which the X are the digits that might be either 1 or 0 and O is Always 0, in which it can verify all of the output data from Table 1. Taking some of the values from Table 1 as an Example

7FFE	0111	1111	1111	1110
A57C	1010	0101	0111	1100
3110	0011	0001	0001	0000
CF06	1100	1111	0000	0110
5A8c	0101	1010	1000	1100
15 bits	XXXX	XXXX	XXXX	XXXO

Comparing all the values from the table with the 15 bits to verify the results as 15 bits of Word Length.

Figure 8 shows the result of 120 generations for the optimization of the Twiddle Factor of 15 bits word length, from Figure 8, it show that the highest value for the optimization for the SNR is 66.917dB which is found in 93rd generation the figure also shows that the GA is able to find the first acceptable solution in the 2nd generation of 63.889dB and the last acceptable solution for this optimization is 63.484dB which is found in the 120th generation. The improvement that was obtained for the 15 bits of word length is (6.21%).

Table 1: Highest Twiddle Factor acceptable solutions for WL 15

Highest Twiddle Factor acceptable solutions for WL 15		
No.	**Real**	**Imaginary**
1	**7ff4**	**000e**
2	**7ff8**	**0006**
3	**7ffe**	**0004**
4	**7ff4**	**0016**
5	**7ff0**	**000e**
6	**764a**	**cf06**
7	**5aa0**	**a596**
8	**3102**	**89c8**
9	**7ffe**	**0016**
10	**5a98**	**a582**
11	**0008**	**8000**
12	**a584**	**a592**
13	**7ffc**	**0006**
14	**3104**	**89c6**
15	**a582**	**a58a**
16	**89d8**	**310a**

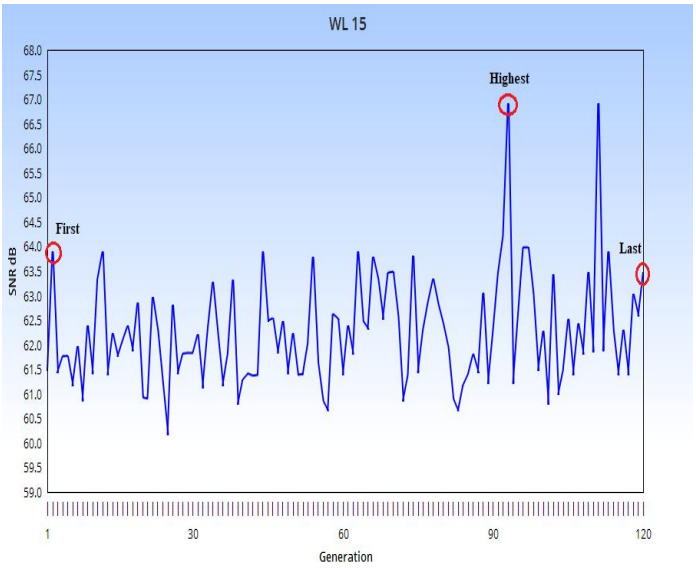

Figure 8:SOGA results for SNR of 15 bits Twiddle Factor

4.1.2 SOGA results for word length of 14 bits

Table 2 shows the Twiddle Factor results that were used to find the highest SNR value which is shown in Figure 9. a 14 bits of Word Length means XXXX XXXX XXXX XXOO of digits, in which the X are the digits that might be either 1 or 0 and O is Always 0, in which it can verify all of the output data from Table 2.

Comparing all the values from the table with the 14 bits to verify the results as 14 bits of Word Length.

Table 2: Highest Twiddle Factor acceptable solutions for WL 14

Highest Twiddle Factor acceptable solutions for WL 14		
No.	**Real**	**Imaginary**
1	**7fe8**	**1c**
2	**7fe8**	**0004**
3	**7ff8**	**0004**
4	**7ff4**	**0014**
5	**7fec**	**000c**
6	**7648**	**cf04**
7	**5a80**	**a57c**
8	**30f8**	**89c0**
9	**7ff0**	**000c**
10	**5a9c**	**a57c**
11	**0008**	**8000**
12	**a57c**	**a590**
13	**7ff8**	**0000**
14	**3110**	**89c0**
15	**a57c**	**a58c**
16	**89d8**	**310c**

Figure 9 shows the results of 120 generations for the optimization of the Twiddle Factor of 14 bits word length. The Figure shows that after decreasing the word length to 14 bits, a value of 66.667dB in the 43rd generation is obtained as the highest acceptable solution, the first acceptable solution has a value of 63.597dB and it is found in the 2nd generation, the last acceptable solution has a value of 65.179dB and it is found in the 119th generation. When comparing between both results for 15 bits and 14 bits it shows that there is a small drop in the highest SNR value, in which it happens because of decreasing the word length from 15 bits to 14 bits which affects the power for the computation of FFT. The improvement that was obtained for the 14 bits of word length is (5.82%).

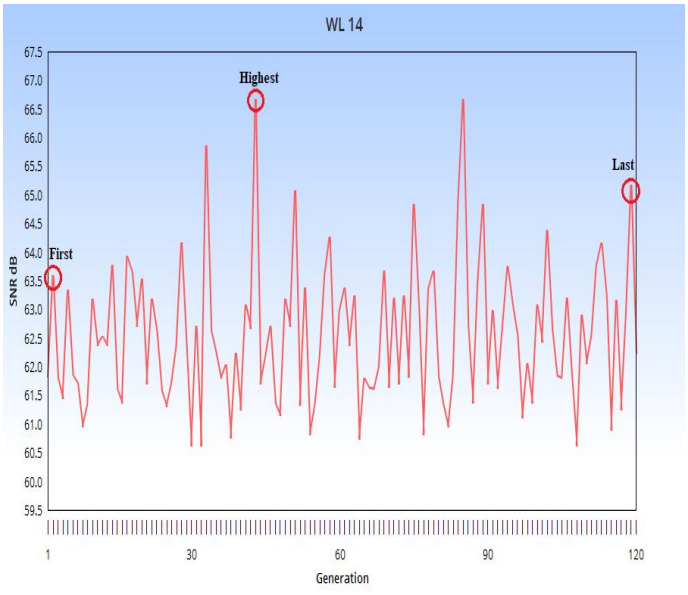

Figure 9: SOGA results for SNR of 14 bits Twiddle Factor

4.1.3 SOGA results for word length of 13 bits

Table 3 shows the Twiddle Factor results that were used to find the highest SNR value which is shown in Figure 10. a 13 bits of Word Length means XXXX XXXX XXXX XOOO of digits, in which the X are the digits that might be either 1 or 0 and O is Always 0, in which it can verify all of the output data from Table 3. Comparing all the values from the table with the 13 bits to verify the results as 13 bits of Word Length.

Table 3: Highest Twiddle Factor acceptable solutions for WL 13

Highest Twiddle Factor acceptable solutions for WL 13		
No.	**Real**	**Imaginary**
1	**7fe8**	**0008**
2	**7ff8**	**0000**
3	**7ff8**	**0010**
4	**7ff8**	**0000**
5	**7fe0**	**0008**
6	**7640**	**cf00**
7	**5a98**	**a590**
8	**30f8**	**89b8**
9	**7ff8**	**0000**
10	**5a98**	**a590**
11	**0008**	**8000**
12	**a598**	**a580**
13	**7ff0**	**0000**
14	**3110**	**89b8**
15	**a590**	**a590**
16	**89c8**	**30f8**

Figure 10 shows the results of 120 generations for the optimization of the Twiddle Factor of 13 bits word length. This figure shows that by decreasing the word length to 13 bits resulted with obtaining 66.645dB as the highest the value for the highest acceptable solution in the 75th generation, the first acceptable solution has a value of 63.608dB and it is found in the 1st generation, the last acceptable solution has a value of 63.207dB and it is found in the 116th generation, by comparing it to the 14 bits, SOGA was able to maintain a good solution for the SNR Value**.** The improvement that was obtained for the 13 bits of word length is (5.78%).

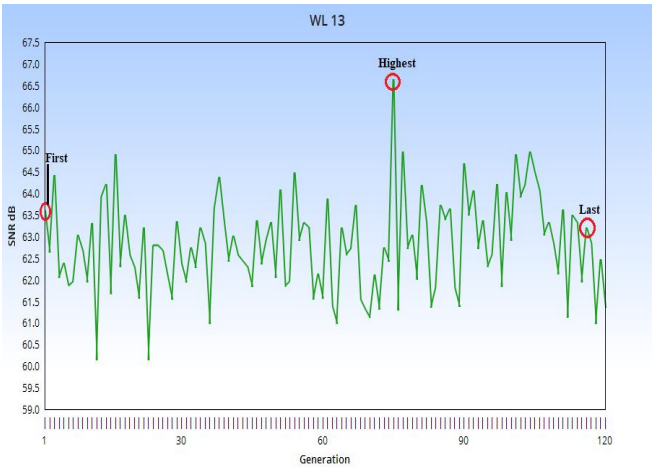

Figure 10 :SOGA results for SNR of 13 bits Twiddle Factor

4.1.4 SOGA results for word length of 12 bits

Table 4 shows the Twiddle Factor results that were used to find the highest SNR value which is shown in Figure 11. a 12 bits of Word Length means XXXX XXXX XXXX OOOO of digits, in which the X are the digits that might be either 1 or 0 and O is Always 0, in which it can verify all of the output data from Table 4.

Comparing all the values from the table with the 12 bits to verify the results as 12 bits of Word Length.

Table 4: Highest Twiddle Factor acceptable solutions for WL 12

Highest Twiddle Factor acceptable solutions for WL 12		
No.	**Real**	**Imaginary**
1	**7ff0**	**0010**
2	**7ff0**	**0000**
3	**7ff0**	**0000**
4	**7fe0**	**0000**
5	**7fe0**	**0000**
6	**7640**	**cf10**
7	**5a80**	**a580**
8	**3100**	**89c0**
9	**7ff0**	**0000**
10	**5a90**	**a580**
11	**0010**	**8000**
12	**a570**	**a580**
13	**7ff0**	**0000**
14	**30f0**	**89b0**
15	**a570**	**a570**
16	**89c0**	**30f0**

Figure 11 shows the results of 120 generations for the optimization of the Twiddle Factor of 12 bits word length, the highest result is 66.431dB in the 35th generation and also, most of the results were more than 63dB which are still considered as acceptable solutions, the first acceptable solution has a value of 63.202dB and it is found in the 1st generation, the last acceptable solution has a value of 63.848dB and it is found in the 120th generation. The improvement that was obtained for the 12 bits of word length is (5.44%).

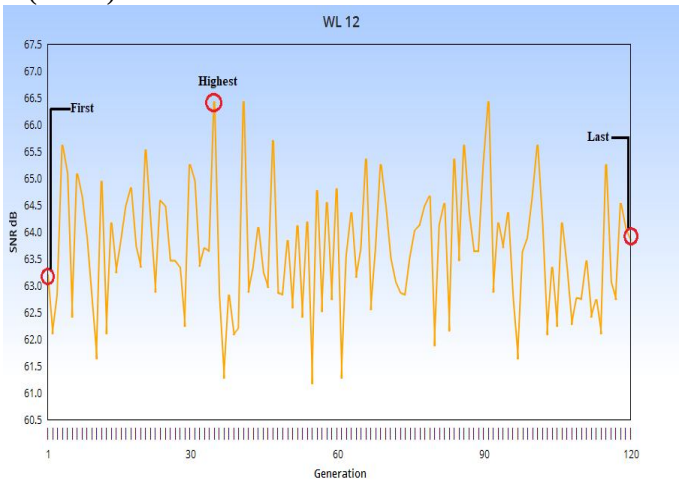

Figure 11 :SOGA results for SNR of 12 bits Twiddle Factor

4.1.5 SOGA results for word length of 11 bits

Continuing with the procedure of lowering the word length to 11 bits for 120 generations, and noticing the results shown in Figure 12, none of the results that showed up is more than 63dB and the highest value that has been found is 61.268dB in the 9th generation. Thus, the GA could not obtain the results that meet with the requirements of the objective goal anymore, therefore the GA optimization has stopped at 11 bits of word length.

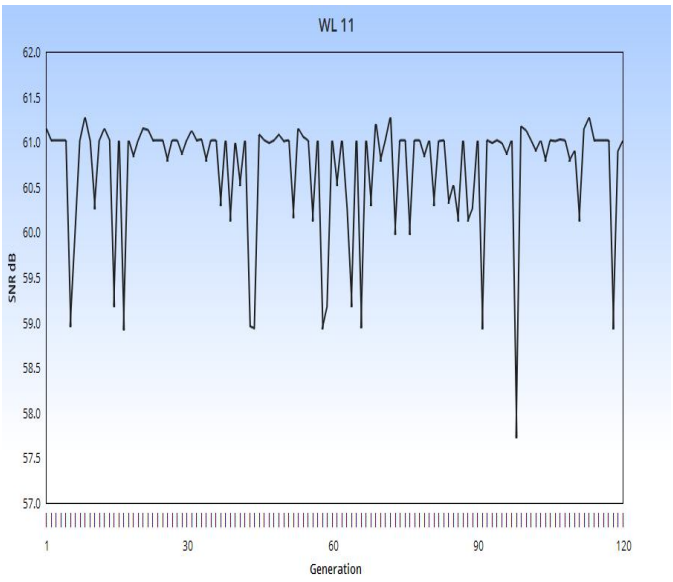

Figure 12: SOGA results for SNR of 11 bits Twiddle Factor

4.1.6 Summary of SNR results for SOGA

The GA for the SOGA has been set to optimize the Word Length of the Twiddle Factor for the FFT processor by using the results from the SNR values, the goal is set to get values for the SNR that are more than 63, the values are decreased for the SNR while decreasing word length for the Twiddle Factor from 15 to 11. The acceptable solutions that were found are summarized and listed in Table 5 for the word length of 15 bits, 14 bits, 13 bits, 12 bits and 11 bits, the results for the 11 bits will be ignored because it did not achieve the required goals for SNR. By analyzing the SOGA results from in Table 5 SOGA managed to find acceptable solutions that keeps the SNR improvement between 6.21% to 5.44% for Word Lengths from 15 bits to 12 bits.

Table 5 : The improvement of SNR for different Word Length Values

Word length	**Target SNR (dB)**	**SNR (dB)**	**SNR Improvement (%)**
15 bits	**63**	**66.917**	**6.21%**
14 bits	**63**	**66.667**	**5.82%**
13 bits	**63**	**66.645**	**5.78%**
12 bits	**63**	**66.431**	**5.44%**
11 bits	**63**	**61.268**	**Not Acceptable Reduction by 2.7%**

4.2 Multi-Objective GA Results

The purpose of the MOGA with the weighted sum approach is to optimize two values of the FFT processor by finding the optimum value for both SNR and SA, which will optimize the effectiveness for the Twiddle Factor for the FFT. In order to do that the MOGA will look for the best objective solutions for the SNR which must meet two goal of objectives, the first goal is the SNR value must be higher than 63dB, as for the second goal, the SA values to look for must have a value less than 192, once these 2 goals achieved these values will be considered as acceptable solutions. the default word length for the FFT is 16 bits, the MOGA will start searching from 15 and below to get values that meet the SNR and SA goals, by decreasing the bits by 1 bit each time and check how many acceptable solutions that can be found, continuing with this procedure until it cannot be able to find a value more than 63dB.

4.2.1 Results for word length of 15 bits

Table 6 shows the Twiddle Factor results that were used to find the highest SNR value which is shown in Figure 4.10. a 15 bits of Word Length means XXXX XXXX XXXX XXXO of digits, in which the X are the digits that might be either 1 or 0 and O is Always 0, in which it can verify all of the output data from Table 4.6

Comparing all the values from the table with the 15 bits to verify the results as 15 bits of Word Length.

Table 6: Highest Twiddle Factor acceptable solutions for WL 15

Highest Twiddle Factor acceptable solutions for WL 15		
No.	**Real**	**Imaginary**
1	**7ffe**	**0000**
2	**7fee**	**0008**
3	**7ff4**	**0000**
4	**7ff6**	**000e**
5	**7ffe**	**000e**
6	**7652**	**cf08**
7	**5aa0**	**a584**
8	**3116**	**89be**
9	**7ff6**	**0006**
10	**5a88**	**a58c**
11	**0004**	**8004**
12	**a586**	**a588**
13	**7ffa**	**0000**
14	**3112**	**89c8**
15	**a58a**	**a58c**
16	**89d4**	**3108**

Figure 13 shows the results of 120 generations for the optimization of the Twiddle Factor of 15 bits word length. The figure shows that the highest acceptable solution value for this optimization for the SNR is 66.442dB in the 75th generation with a SA value of 178, the first acceptable solution is found in the 11th generation with a value 63.774dB of for SNR and SA value of 180, and the last acceptable solution is found in the 120th generation with a value of 63.013dB for SNR and SA value of 180,the improvement that was obtained for the SNR is (5.46%) and a reduction by (7.29%) for the SA.

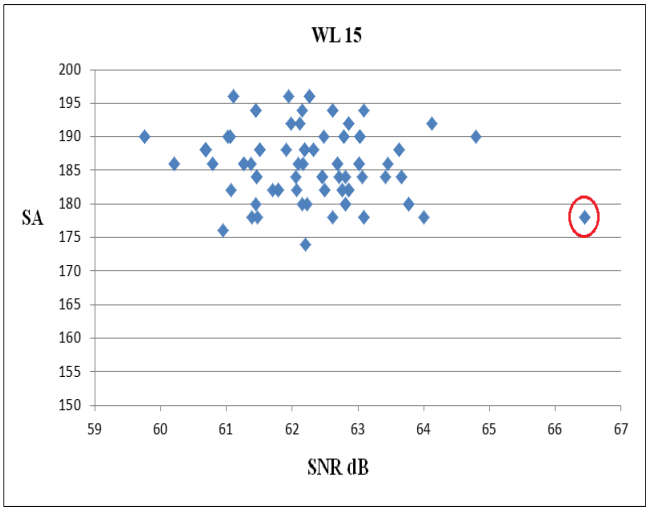

Figure 13: MOGA results for SNR and SA of 15 bits Twiddle Factor

4.2.2 Results for word length of 14 bits

Table 7 shows the Twiddle Factor results that were used to find the highest SNR value which is shown in Figure 13. a 14 bits of Word Length means XXXX XXXX XXXX XXOO of digits, in which the X are the digits that might be either 1 or 0 and O is Always 0, in which it can verify all of the output data from Table 7.

Comparing all the values from the table with the 14 bits to verify the results as 14 bits of Word Length

Table 7: Highest Twiddle Factor acceptable solutions for WL 14

Highest Twiddle Factor acceptable solutions for WL 14		
No.	**Real**	**Imaginary**
1	**7fec**	**0014**
2	**7ff4**	**0018**
3	**7ff8**	**0010**
4	**7fec**	**0010**
5	**7fe4**	**0010**
6	**7648**	**cf14**
7	**5a80**	**a58c**
8	**3100**	**89c4**
9	**7ff0**	**0000**
10	**5a94**	**a57c**
11	**0000**	**8008**
12	**a590**	**a584**
13	**7ff0**	**0000**
14	**3104**	**89d8**
15	**a590**	**a580**
16	**89c4**	**30fc**

Figure 13 shows the results of 120 generations for the optimization of the Twiddle Factor of 14 bits word length. The Figure shows that after decreasing the word length to 14 bits was obtained the highest acceptable solution of value 66.014dB in the 81st generation for the SNR with a SA value of 172, which provides an improvement of (4.78%) for the SNR and a reduction of (10.41%) for the SA, the first acceptable solution is found in the 3rd generation with a value 63.489dB of for SNR and SA value of 166, and the last acceptable solution is found in the 117th generation with a value of for 64.349dB SNR and SA value of 168.

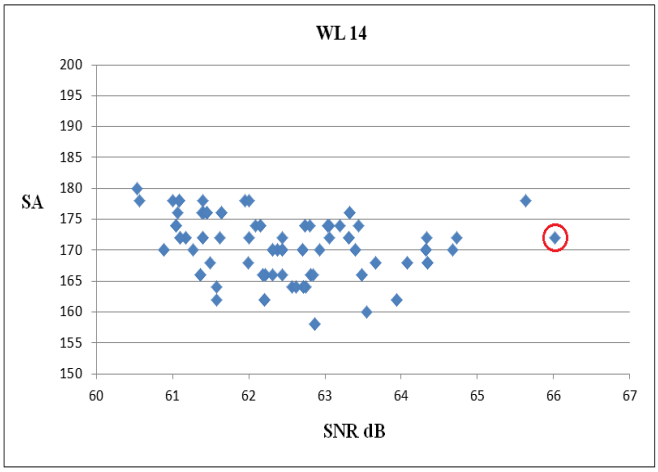

Figure 13 :MOGA results for SNR and SA of 14 bits Twiddle Factor

4.2.3 Results for word length of 13 bits

Table 8 shows the Twiddle Factor results that were used to find the highest SNR value which is shown in Figure 14. a 13 bits of Word Length means XXXX XXXX XXXX XOOO of digits, in which the X are the digits that might be either 1 or 0 and O is Always 0, in which it can verify all of the output data from Table 8.

Comparing all the values from the table with the 13 bits to verify the results as 13 bits of Word Length.

Table 8: Highest Twiddle Factor acceptable solutions for WL 13

Highest Twiddle Factor acceptable solutions for WL 13		
No.	**Real**	**Imaginary**
1	**7ff8**	**0010**
2	**7ff0**	**0010**
3	**7ff8**	**0008**
4	**7fe0**	**0000**
5	**7ff0**	**0018**
6	**7648**	**cf10**
7	**5a98**	**a578**
8	**30f8**	**89c0**
9	**7ff8**	**0018**
10	**5a88**	**a578**
11	**0000**	**8010**
12	**a578**	**a588**
13	**7ff8**	**0008**
14	**30f8**	**89c0**
15	**a590**	**a590**
16	**89d0**	**3110**

Figure 14 shows the results of 120 generations for the optimization of the Twiddle Factor of 13 bits word length. This figure shows that by decreasing the word length to 13 bits was obtained the highest value of 65.976dB in the 70th generation with a SA value of 152, the overall improvement for the optimization is (4.72%) for the SNR and a reduction by (20.83%) for the SA, the first acceptable solution is found in the 6th generation with a value of 63.866dB for SNR and SA value of 160, and the last acceptable solution is found in the 120th generation with a value of 65.892dB for SNR and SA value of 152. by comparing it to the 14 bits which shows that SOGA was able to maintain a good solution for the SNR Value.

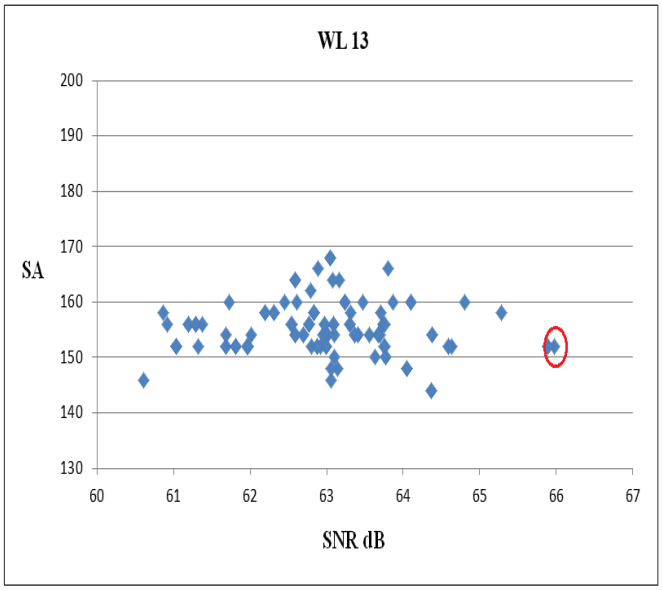

Figure 14 :MOGA results for SNR and SA of 13 bits Twiddle Factor

4.2.4 Results for word length of 12 bits

Table 9 shows the Twiddle Factor results that were used to find the highest SNR value which is shown in Figure 15. a 12 bits of Word Length means XXXX XXXX XXXX OOOO of digits, in which the X are the digits that might be either 1 or 0 and O is Always 0, in which it can verify all of the output data from Table 9.

Comparing all the values from the table with the 12 bits to verify the results as 12 bits of Word Length.

Table 9: Highest Twiddle Factor acceptable solutions for WL 12

Highest Twiddle Factor acceptable solutions for WL 12		
No.	**Real**	**Imaginary**
1	**7ff0**	**0000**
2	**7ff0**	**0000**
3	**7ff0**	**0010**
4	**7fe0**	**0000**
5	**7ff0**	**0000**
6	**7650**	**cf10**
7	**5a80**	**a570**
8	**30f0**	**89c0**
9	**7ff0**	**0000**
10	**5a90**	**a580**
11	**0000**	**8000**
12	**a570**	**a570**
13	**7ff0**	**0010**
14	**3100**	**89c0**
15	**a570**	**a570**
16	**89c0**	**30f0**

Figure 15 shows the results of 120 generations for the optimization of the Twiddle Factor of 12 bits word length, the highest result is 65.801dB in the 52th generation for the SNR with a SA value of 146, also many of the results were more than 63dB which are still considered as acceptable solutions and the improvement for the optimization is (4.44%) for the SNR and reduction by (23.95%) for the SA, the first acceptable solution is found in the 1st generation with a value of 64.116dB for SNR and SA value of 144, and the last acceptable solution is found in the 120th generation with a value of 63.757dB for SNR and SA value of 140.

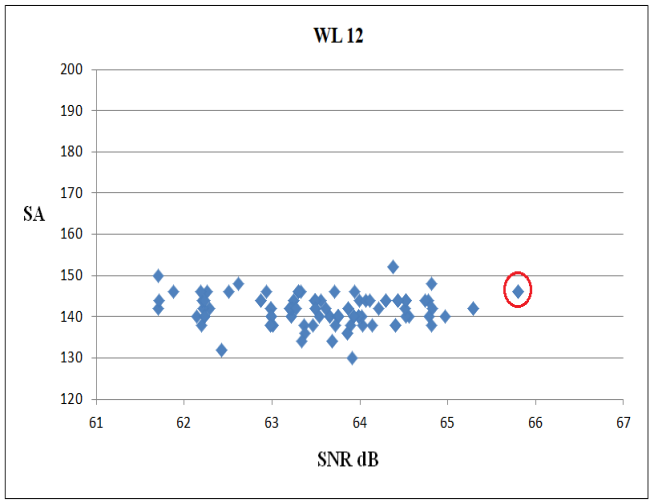

Figure 15 :MOGA results for SNR and SA of 12 bits Twiddle Factor

4.2.5 Results for word length of 11 bits

By continuing to lower the word length to 11 bits for 120 generation, and noticing the results which are shown in Figure 16 none of the results that showed up is more than 63dB and the highest value that can be found is 62.381dB in the 23rd generation. Thus, the GA could not get the results that meet the requirement goal anymore. Therefore the GA optimization has stopped at 11 bits of word length.

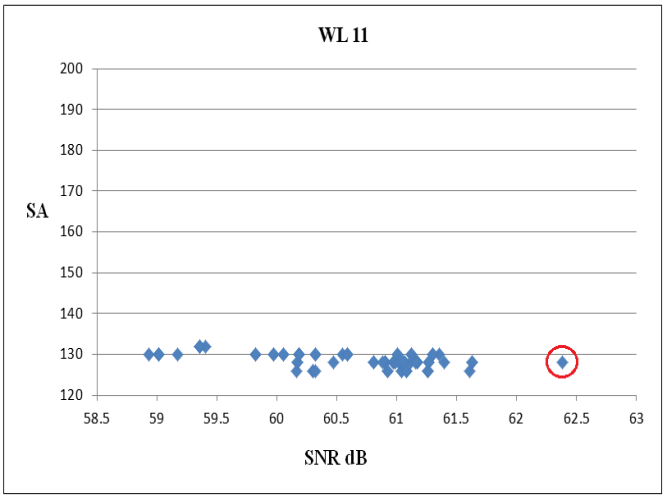

Figure 16 :MOGA results for SNR and SA of 11 bits Twiddle Factor

4.3 Summary of SNR and SA results for MOGA

The GA for the MOGA has been set to optimize the Word Length of the Twiddle Factor for the FFT processor by achieving two objectives at the same time the first objective is to get a SNR value higher than 63dB and the second objective is to get a SA lower than 192. The values are decreased for the SNR while decreasing the word length for the Twiddle Factor from 15 to 11, however, the results for SA value keeps improving while lowering the word length. The acceptable solutions were summarized and listed in Table 4 for the word length of 15 bits, 14 bits, 13 bits and 12 bits. By analyzing the results, the MOGA was able to maintain an improvement for SNR from (5.46% to 4.44%) and SA reduction from (7.29% to 23.95%) when lowering the word length from 15 bits to 12 bits

Table 10:4 The improvement of SNR and SA for different Word Length values

WL	**SNR (dB)**	**SA**	**SNR Improvement (%)**	**SA Reduction (%)**
15 bits	**66.442**	**178**	**5.46**	**7.29**
14 bits	**66.014**	**172**	**4.78**	**10.41**
13 bits	**65.976**	**152**	**4.72**	**20.83**
12 bits	**65.801**	**146**	**4.44**	**23.95**

5. CONCLUSION

The research is set to design a reconfigurable FFT Twiddle Factor of 16-point Radix 4 Single Path Delay Feedback by creating a Single Objective Genetic Algorithm (SOGA) while being implemented on the Raspberry Pi. The SOGA is set only to find the best possible solutions for the Signal to Noise Ratio (SNR), the targeted solutions must be higher than 63dB while lowering the Word Length. The proposed work managed to lower the word length to 12 bits while having an improvement of SNR between (5.75% and 5.47) for 15 to 12 bits of word length.. By getting the required solutions the research can successfully meet the research objectives of reducing the power consumption for the FFT processor and by using the Raspberry Pi (RPi), the research can be implemented on low cost mini-computer which can be considered as efficient as using a fully functional computer because it can give the same results as the fully functional computer.

The Next step was to set the research for the MOGA to find the solutions for both SNR and SA which must meet the goal of being higher than 63dB for the SNR and lower than 192 for the SA. The proposed work managed to lower to 12 bits while having an improvement of SNR between (5.61% and 4.2%) and SA reduction between (7.29 and 30.2) while lowering the Word Length from 15 to 12. By getting the required solutions the research can successfully meet the research objectives of reducing the power consumption for the FFT processor and by using the Raspberry Pi (RPi), the research has been implemented on low cost mini-computer which can be considered as efficient as using a fully functional computer because it can give the same results as the fully functional computer.

REFERENCES

[1] Pang Jia Hong, Nasri Sulaiman. (2010). Genetic Algorithm Optimization for Coefficient of FFT Processor. In *Australian Journal of Basic and Applied Sciences*, 4(9): pp 4184-4192.

[2] Nasri S. Tughrul A. (2004) A Genetic Algorithm for the Optimisation of a Reconfigurable Pipelined FFT Processor, Department of Electronics and Electrical Engineering the University of Edinburgh Scotland.

[3] Franco Busetti. (2001). Genetic algorithms overview.

[4] K. Deb. (2001). Multi-Objectives Optimization Using Evolutionary Algorithms. John Wiley and Sons.5-Evolutionary Feature Subset selection for Pattern Recognition applications by George Papakostas and Dimitrios E. Koulouriotis

[5] Manimaran A.*and S.K. Sudeer. (2016). A Novel VLSI Based Pipelined Radix-4 Single-Path Delay Commutator (R4SDC) FFT.

[6]M.Tawarish, Dr. K. Satyanarayana A Review onb PricingPrediction on Stock Market by Different Techniques in the Field of Data Mining and Genetic Algorithm International Journal of Advanced Trends in Computer Science and Engineering Volume 8 No. 1 (2019) https://doi.org/10.30534/ijatcse/2019/05812019

[7]M.Tawarish, Dr. K. Satyanarayana An enabling technique analysis in Data Mining for Stock Market trend by Approaching Genetic Algorithm, International Journal of Advanced Trends in Computer Science and Engineering Volume 8 No. 1 (2019)

[8] N. Srinivas and Kalyanmoy Deb. (1994) .Multi-objective Optimization Using Nondominated Sorting in Genetic Algorithms. In *Journal of Evolutionary Computation*, Vol. 2, No3, pp 221-248.

[9] K. Deb. (2001). Multi-Objectives Optimization Using Evolutionary Algorithms. John Wiley and Sons.

[10] G.A. Papakostas, D.E. Koulouriotis, A.S. Polydoros and V.D. Tourassis, April 2011 Evolutionary Feature Subset selection for Pattern Recognition applications. Democritus University of Thrace, Department of Production Engineering and Management,Greece. https://doi.org/10.5772/15655

[11] Mahmud B. and Masuri O. (2009), Hardware Implementation of a Genetic Algorithm Based Canonical Singed Digit Multiplierless Fast Fourier Transform Processor for Multiband Orthogonal Frequency Division Multiplexing Ultra Wid Journal of Mathematics and Statistics 5 (4): 241-250, 2009 ISSN 1549-3644 Science Publications.

[12] Mitsuo Gen, Runwei Cheng. (2000). Genetic Algorithm & Engineering Optimization. John Wiley and Sons.

[13] N. Srinivas and Kalyanmoy Deb. (1994).Multi-objective Optimization Using Nondominated Sorting in Genetic Algorithms. In Journal of Evolutionary Computation, Vol. 2, No3, pp 221-248. https://doi.org/10.1162/evco.1994.2.3.221

[14] D. Flandre, M.Debleser, A.Vandemeulebroecke, P.G.A Jesper. (1988). Comparison of Equivalent Precicion Dedicated FFT Processors. In Circuits and Systems IEEE International Symposium, volume 2, pp 1919-1922.

[15] Nasri S. and Tughrul A. (2005),A Multi-objective Genetic Algorithm for On-chip Real-time Optimisation of Word Length and Power Consumption in a Pipelined FFT Processor targeting a MC-CDMA Receiver. School of Engineering and Electronics The University of Edinburgh, King's Buildings