On the filtering properties of evolved gate arrays

Julian F. Miller
School of Computing, Napier University
219 Colinton Road, Edinburgh, EH14 1DJ, UK
j.miller@dcs.napier.ac.uk

Abstract

A small gate array is evolved extrinsically to carry out a low pass filtering task defined over fifteen different frequencies. The circuit is evolved by assessing its response to digitised sine waves. Two different fitness functions are contrasted. One is based on computing the sum of the absolute differences between the actual response and that desired, the other is defined by examining characteristics of the Discrete Fourier Transform of the output. The gate arrays possess some linear properties, which means that they are capable of filtering composite signals which have not been encountered in training. This includes signals with noise added and with frequencies which are not in the training set.

1 Introduction

Recently it has been demonstrated how a digital circuit may be evolved in-circuit to carry out a frequency discrimination task [1] [21][22]. The behaviour of the circuit was intimately related to the physical properties of the actual integrated circuit used (i.e. temperature, silicon batch). The circuit was not clocked and the evolutionary algorithm had to allow the circuit to construct internal timing circuits with periods varying over many orders of magnitude. In this paper it is shown that by using digitised samples of an incoming signal one can evolve a gate array to carry out a low pass filtering task involving fifteen frequencies. The gate array is feed-forward and requires no clock. The method is novel in that it does not use one of the fundamental paradigms of conventional digital signal processing, namely, the difference equation [10]. The latter is formally represented in the following way. The output of the filter at time *n*, *y(n)*, may be a function of *N* samples of the signal *x(n-i)* at earlier times (FIR: finite impulse response), and may also, if feedback is present (IIR: infinite impulse response), involve earlier outputs *y(n-i)* given by the following equation:

$$y(n) = \sum_{i=0}^{N-1} a_i x(n-i) + \sum_{i=1}^{M} b_i y(n-i) \qquad (1)$$

where the coefficients a_i and b_i are real valued floating-point numbers. The essential problem of filter design is the choice of $\{a_i\}$, $\{b_i\}$, *N*, and *M*, so that the filter has the desired behaviour (i.e. frequency response). In practice the coefficients $\{a_i\}$, $\{b_i\}$ are of finite precision. The practical requirements of implementing such a system in hardware consists of providing a number of shift registers, multipliers, and adders. Large bit multipliers are very costly in hardware terms. Three of the most important factors in the design of digital filters are quality of signal response, size (cost) of hardware implementation, and speed of operation. There are many traditional approaches which have been developed to address these issues [10]. In particular one popular method for reducing the implementational complexity is to restrict the filter coefficients to integer coefficients, see [6] and references therein. Recently, researchers have started to explore the application of evolutionary algorithms to filter design [1][3][4][5][7][8][18][19][20][23]. The essential idea employed by most of these authors is to use an evolutionary algorithm to optimise the filter coefficients. This may be in combination with finite wordlength analysis [1][8] for IIR filter design, or it may be in an adaptive context [7][20]. Other workers have employed evolutionary algorithms to optimise coefficients together with add and shift operations in so-called multiplier-less designs [18][19][23].

The pioneering concept of gate-level evolution of digital functions was developed in [9]. In [17] the authors generalised the concept of gate-level evolution to the so-called functional level, and they showed how it was possible to carry out adaptive equalisation on a communications channel with superior bit error rates to the conventional least mean squares method. One of the objectives of this paper is to show that that the possibilities afforded by *gate-level* evolution have been left largely unexplored, and that there remains much fundamental work to be done at this level. An additional motivation for attempting this work is

[1] The frequency discrimination task consisted of distinguishing between square wave signals of frequencies 1kHz and 10kHz.

the enormous potential for new knowledge discovery afforded by the simple nature of logic functions. In other words, can new principles be extracted from gate-level evolution that can inspire and contribute to new methodological paradigms? There are of course fundamental questions which need to be addressed if such a filtering method is to become practicable. Foremost among these would be the question of linearity. If a gate array is to be trained to carry out a filtering task, then can this be done in such a way that composite signals, which can be represented as weighted sums of sine waves, will also be filtered? This would imply that the circuit is, at least, weakly linear. This has recently been shown to be possible [15][16] and once again the results presented here (section 4) confirm this fact. The fitness of the gate arrays is assessed by presenting a number of digitised sine waves to the gate array. In earlier work the fitness function was related to the frequency response of the gate array by taking the Discrete Fourier Transform (DFT) of the output. Although the output signals of the gate array were reasonably well behaved in the frequency domain there was noticeable distortion in the time domain. In this paper an attempt is made to improve the time domain behaviour by defining another fitness function which looks at the sum of the differences between the circuit output and that desired. The new behaviour of the fitness function is compared with the original. In addition the response of the evolved filters to noisy signals and signals with non-integral frequencies is examined. Again the findings are encouraging as some of the filters are noise tolerant and are able to respond appropriately to non-integral frequencies.

The actual method employed here to evolve a gate array is developed from earlier work in [11-14]. This is explained in section 2. In that work [12-14], the objective was to synthesise an entire truth table. This becomes increasingly time consuming and difficult as the number of inputs grow. It is obvious that attempting to evolve truth tables of larger sizes will not be feasible. It was argued in [11] that the real applications for gate-array evolution probably lie in real number mapping problems, where the digitised real numbers are presented to a circuit and a digitised real number output is desired. In such a scenario the number of input conditions is determined by the problem and is not necessarily an exponential function of the number of inputs. Such a scenario is ideally furnished by the digital filtering task and this is explained in section 3. In section 4 the evolved filtering characteristics of the gate arrays are presented, including some results which show the quasi-linear behaviour. Some hardware aspects of the evolved circuits are discussed in section 5. Conclusions are given in section 6.

2 Gate-level evolution of digital circuits

The chromosome representation can be explained with a simple example. Fig. 1 shows a small gate array consisting of four logic cells. The logic cells in this case have functions XOR, AND, or MUX (multiplexer). The gate array implements the one-bit adder (with carry-in). The circuit in question actually arose in an earlier experiment reported elsewhere [14] and is quite novel in its own right. A, B, and Cin represent the primary inputs. Cout and Sum are the output bits of the adder. Each cell is assumed to possess three input connections. If the cell function does not require inputs then the corresponding genes are ignored. For example the upper right cell (output 5) below has input connections 3, 2, 1. This means that the first input is connected to the output of the cell with output label 3 (upper left), the second input is connected to the primary input Cin, and the third input is connected to primary input B. The functionality of each cell is expressed as the fourth gene associated with each cell. The primary outputs of the gate array are also expressed as connections. For example Cout is connected to the output of the cell with output label 6. The gate array is envisaged as being divided into vertical columns of cells and the representation is so constrained that columns of cells may only have their inputs connected to connection points on their left. This ensures the feed-forward nature of the circuit and removes any time dependent behaviour. Actually the connectivity is further constrained by the presence of a parameter denoted *l*, which dictates the number of columns on the left (including the primary inputs at column zero) to which the inputs of cells in column *l* may be connected. The purpose of this is to constrain the fan-out of signals and thereby improve the ease with which the circuit may be routed when it is physically implemented.

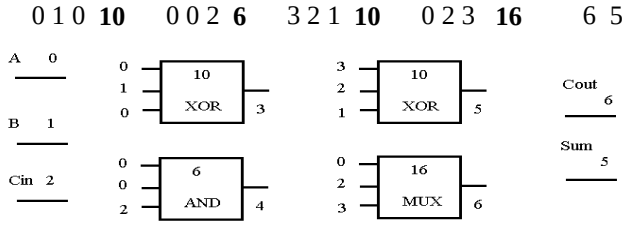


Figure 1: Genotype and phenotype for the gate array of logic cells which implement a one-bit adder (with carry in)

The allowed cell functions can be chosen to be any subset of those shown in Table 1, where ab implies a AND b, $\bar{a}$ indicates NOT a, $\oplus$ represents the exclusive-OR operation and + the OR operation. Functions 0-15 are the basic binary functions of 0, 1 and two inputs. Functions 16-19 are all binary multiplexers with various inputs inverted, function 20 describes a Reed-Muller universal logic module (ULM).

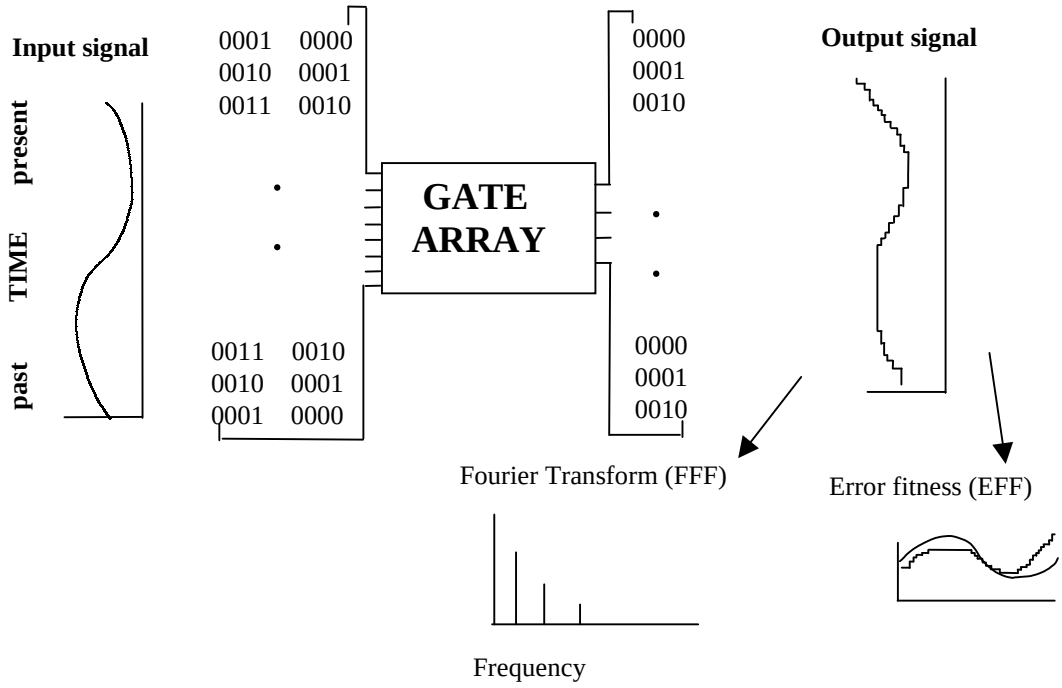


Figure 2: The training scenario for evolving a gate array to have filtering properties

It is important to note that one can consider multiplexers to be *atomic* both formally and from an implementational point of view. It is atomic in that it is a ULM so that it can be used to represent any logic function. Additionally some modern FPGAs now use a multiplexer based architecture so that all two input gates are synthesised with multiplexers.

Table 1: Allowed cell functions

0	1	2	3	4	5	6	7	8	9
0	1	a	b	$\bar{a}$	$\bar{b}$	ab	a $\bar{b}$	$\bar{a}$b	$\bar{a}$ $\bar{b}$
10		11		12		13		14	
a ⊕ b		a ⊕ $\bar{b}$		a + b		a + $\bar{b}$		$\bar{a}$ + b	
15		16		17		18		19	
$\bar{a}$ + $\bar{b}$		a $\bar{c}$ + bc		a $\bar{c}$ + $\bar{b}$c		$\bar{a}$ $\bar{c}$ + bc		$\bar{a}$ $\bar{c}$ + $\bar{b}$c	

The evolutionary algorithm employed was a simple form of (1+4) evolutionary strategy (ES). In this case a population of 5 random chromosomes were randomly generated and the fittest chromosome selected. The new population is then filled with mutated versions of this. Random mutation was defined as a percentage of genes in the population which were mutated. The mutation operator respected the feed-forward nature of the circuits and also the different alphabets associated with connections and functions.

3 Evolving a filter response with a gate array

The incoming analogue signals which are to be processed by the gate array are sampled at frequency f_s and s samples are collected. The required total sampling time t is given by s/f_s. The samples are digitised and represented by a wordlength of w bits. In a FIR filter of order n one therefore must collect nw bits at each sampling time. These nw bits for the s samples are collected and represent the input conditions to the gate array. For each nw input bits the gate array must produce w output bits. In this way a set of input-output conditions is defined. The entire arrangement is shown in fig. 2. In this figure an input sine wave is shown on the left which is digitised to binary numbers with $w=4$, and $n=2$. An entire history of samples are collected for each sine wave. These are the input conditions presented to the gate array. On the right of the gate array is shown the outputs of wordlength equal to 4 bits. The input signals chosen were pure sine waves with zero phase. Two fitness functions were defined. In the first when s samples have been collected the discrete Fourier transform (DFT) is taken. A program which was freely available in [10] was used to do this. In this way the frequency characteristics of the evolving gate array can be assessed for each input signal. For brevity we refer to this fitness function as Fourier Fitness Function (FFF). The second fitness function did not carry out a DFT but was

based on the difference between the actual circuit output and the desired output, this is referred to as the Error Fitness Function (EFF). Both these functions are defined more exactly below. The sine waves had frequencies which were integral multiples of the fundamental f_1 ($2\pi f_s/s$) up to the Nyquist frequency[2], f_n (half sampling frequency) minus 1. The sine waves were translated by the addition of a d.c. component so that they assumed only positive values. One can envisage this more clearly by noting that the fundamental corresponds to a single exact sine cycle fitting into the sampling window (fig. 2). The power in the frequency domain $W(f)$, defined as the modulus of the output response in the complex frequency domain, is scaled by dividing by the maximum power associated with the DFT of the fundamental sine wave. The d.c. component of the output is ignored. Define the maximum power over all frequencies, $W_{\max}$, and the maximum power over all frequencies excluding f_i, $W^i_{\max}$,

$$W_{max} = max\{ W(f_j), \forall j: f_1 \le f_j \le f_n - 1\},$$
$$W^i_{max} = max\{ W(f_j), \forall j\ j \neq i: f_1 \le f_j \le f_n - 1\} \quad (2)$$

Define δ_i to be 1 if the frequency i is to be passed, and 0 if it is to be stopped. The fitness is calculated with a user defined set of frequency rewards r_i (fitness profile). The Fourier fitness (FFF) is given by (3) below, where n_+, and n_-, represent the number of frequencies to be passed, and stopped, respectively (over all frequencies up to f_n-1).

$$x_i^F = \delta_i (W(f_i) - W^i_{\max}) \frac{r_i}{n_+} + (1 - \delta_i)(1 - W_{\max}) \frac{r_i}{n_-} \quad (3)$$

The error based fitness (EFF) is calculated in the following way. Define s_j and y_j to be the input signal and circuit output at sample time j respectively. Then the error E is defined as the sum over samples of the absolute difference between s_j and y_j. Y is defined as the sum over samples of y_j. The elementary error-based fitness (EFF) is defined by (4) below,

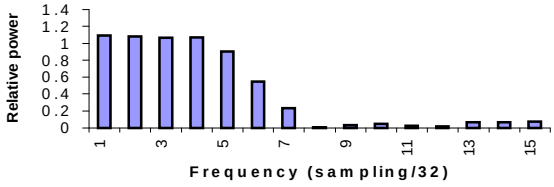
$$x_i^E = \left(\frac{\delta_i}{1 + E/K}\right)\frac{r_i}{n_+} + \left(\frac{1 - \delta_i}{1 + Y/K}\right)\frac{r_i}{n_-} \quad (4)$$

where the scaling constant K is given by $(2^w-1)s$. Note that the definitions given in (3) and (4) are scale invariant and just assign a fixed maximum fitness contribution for the pass regions and stop regions irrespective of their size. The total fitness x^F or x^E associated with a given chromosome is then given by the sum of the components x_i for all frequencies up to f_n-1. These definitions of fitness mean that in the Fourier fitness definition one is trying to suppress all sine waves with frequencies in stop region, and trying to enhance only *pure* frequencies (uncorrupted sine waves) in the pass region. In the error fitness one is trying to minimise the differences between the output and input signals in the pass region, and minimise the total signal in the stop region.

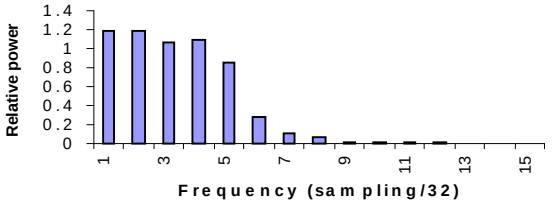
[2] Nyquist's theorem states that in order to reconstruct a signal perfectly half the sampling frequency (known as the Nyquist frequency) must be greater than the highest frequency component in the signal.

4 Results

The experimental parameters for this paper are given below, All filters were evolved with s=32, w =8, filter order = 6. The only allowed gate function for all experiments was a multiplexer (type 16). The population size = 5, mutation rate = 1.00, number generations = 10000. The gate array was 7 rows and 7 columns. The levels back parameter was set to 4. The passband was defined by frequencies f_1-f_5. For each fitness function the best chromosomes of 25 runs of the algorithm were selected. In these experiments the reward profiles r_i were set uniformly to one The frequency response of a filter is defined as the plot of *maximum* output power with frequency. The responses of the evolved filters are shown below (note that the in these figures the frequencies are in angular form).



(i)



(ii)

Figure 3: Frequency responses of evolved low pass gate arrays with EFF (i) and FFF (ii)

In order to understand the complete behaviour of the evolved gate arrays it is necessary to examine how they respond to the individual signals with which they were trained. A perfect linear filter would be transparent to any signal which was totally composed of frequencies in the passband. It would be opaque to frequency components which were in the stopband. It is not possible to assume any

of this when evolving a gate array. It is only possible to assess the quality of the evolved gate arrays by examining how they respond to various signals. In section 4.1 the responses of the fittest evolved gate arrays to pure sine waves are shown. In section 4.2 the filters are presented with signals which are sums of two pure sine waves of varying amplitudes. The fitness functions with which the gate arrays were evolved only looked at responses to pure sine waves with frequencies which were integral multiples of the fundamental. Thus the incident signals are entirely new. In section 4.3 the responses of the filters to a signal which is a sum of three sine waves are given. In section 4.4 noise is added to the fundamental sine wave and the responses of the gate arrays are examined. Finally in section 4.5 the responses to signals with frequencies which are not integer multiples of the fundamental are examined. It is important to realise that all these experiments are necessary in order to assess the true behaviour of the evolved filters. Unlike conventional filter design the filters have not been designed using a linear difference equation and all the power of the mathematics of modern signal processing theory. *Thus we can assume nothing about the behaviour of the evolved filters.*

4.1 Gate array response to pure sine waves

In figs. 4 – 6 the actual responses of the fittest evolved gate arrays to pure sine waves are shown. Fig. 4 shows the time and frequency domain response of the EFF evolved gate array (4 (i), 4(iii)) and the FFF evolved gate array (4 (iv), 4(v)) when presented with the fundamental sine wave.

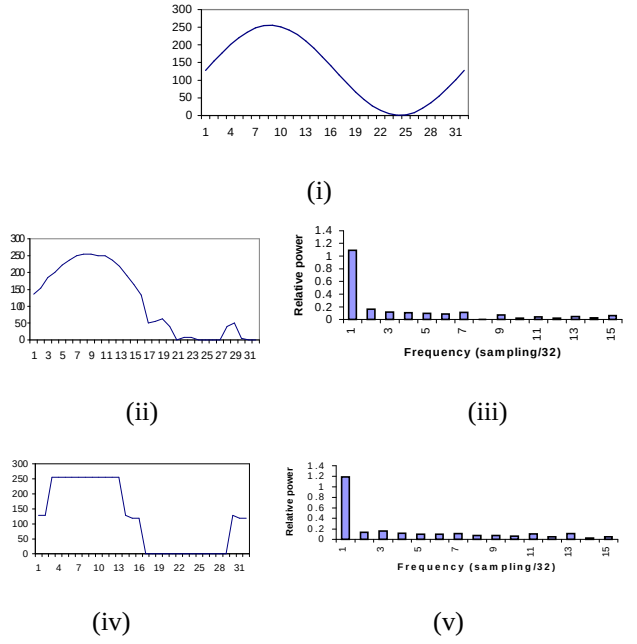


Figure 4: Incident signal f_1 (i), output and frequency responses for EFF (ii,iii) and FFF (iv,v)

In figure 4 (ii) the shape of output signal of the gate array evolved using the EFF is good for half a cycle but doesn't return to the mean level (128) at the end of the sample window (32). The FFF gate array shows better periodic behaviour but has lost much of the shape information of the orginal signal and it is tending to operate at three levels (0, 128 and 255). However the frequency responses appear resonably good in that the output is dominated by the fundamental frequency (figs. 4 (iii) and 4 (v)).

Figs. 5(i) – 5(v) show the gate array responses to a pure sine signal on the edge of the passband. First of all note the distortion in the input signal which is due to the rounding error incurred in integer arithmetic. The response of the EFF gate array (5(ii)) is again reasonably good but it is prone to clipping when the input signal is of very low amplitude. The response in the frequency domain (5(iii)) shows a number of 'off-frequency' components.

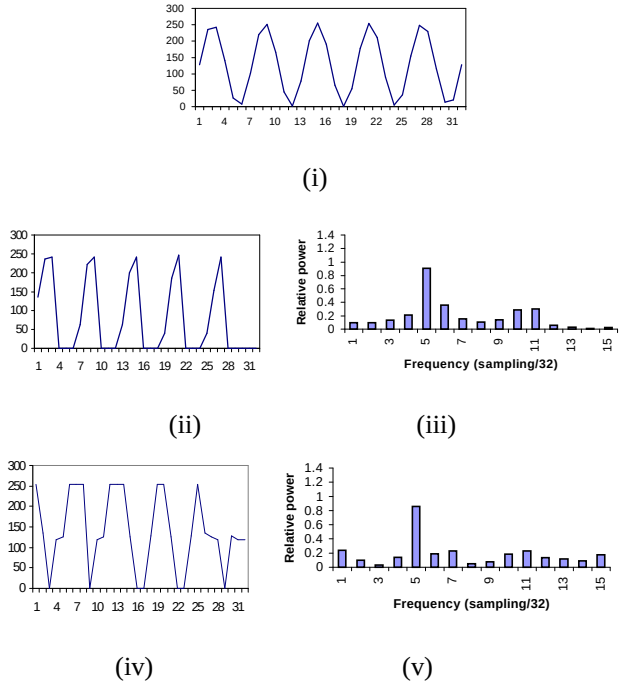
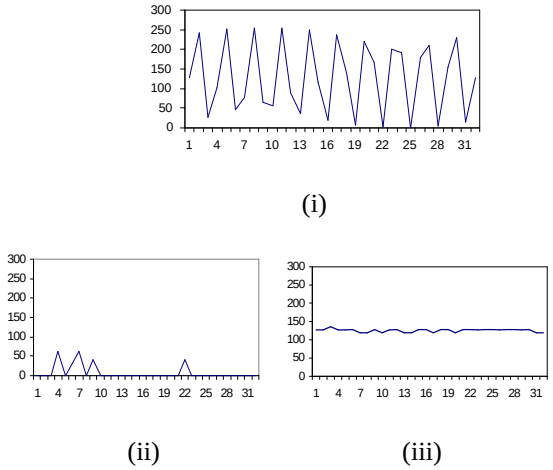


Figure 5: Incident signal f_5 (i), output and frequency responses for EFF (ii,iii) and FFF (iv,v)

Examining (5(iv)) it can be seen how FFF gate array is tending to operate at three levels (as with fig. 4). The frequency response (5(v)) is a little cleaner than with the EFF. The main thing to note is that the input signal is still being transmitted.

Fig. 6(i) shows an incident pure sine wave with a frequency ten times the fundamental. The frequency is well into the stopband. Thus the signal should be highly attenuated by the evolved gate arrays. Figs 6(ii) and 6(iii) show the output response for the EFF and FFF gate arrays respectively. There is very obvious attenuation. Note that

the dc response of 6(iii) is ignored by the FFF (see equations 2 and 3) as there is no time dependent behaviour.



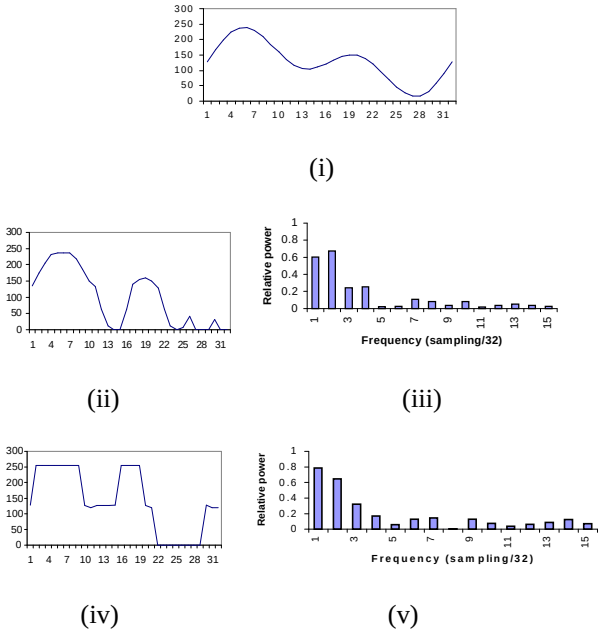
(i)

(ii) (iii)

Figure 6: Incident signal f_{10} (i), and output responses for EFF (ii) and FFF (iii)

4.2 Gate array response to sums of two sine waves

In the next three experiments signals were presented to the evolved gate arrays which were not in the training set. However the signals were sums of two sine waves each of which was present in the training set (albeit with a different amplitude). In two of the experiments the frequencies of the sine waves lay in the passband. In the first of these the sine waves had equal amplitude (fig. 7) equal to 0.5 and frequencies f_1 and f_2.



(i)

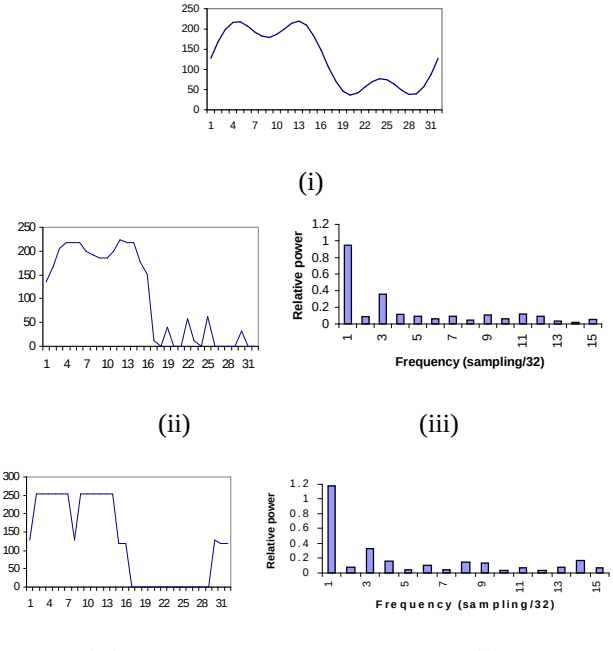
(ii) (iii)

(iv) (v)

Figure 7: Incident signal $0.5(f_1+ f_2)$ (i), output and frequency responses for EFF (ii,iii) and FFF (iv,v)

Examining fig. 7(ii) one can see that the output response using th EFF is quite good. Again it tend to plummet to zero as the input signal drops to 128 and it tends to stay at zero when the input signal rises again from 0 to 128. In the frequency domain the component sine waves in the output signal can be examined (fig 7(iii)) and it is seen that the dominant amplitudes do correspond to the correct frequencies. The amplitudes of these components are not quite equal however and there are other amplitudes not present in the input signal. It is clear that the gate array is behaving in an almost linear manner. The behaviour of the ouput response of the gate array evolved with FFF is also good. It exaggerates the changes in the input signal but correctly identifies the flatter region between samples 12 and 16. It also rises again in the sample region 28-32. Once again the frequency response bears out the nearly linear response as the output signal is dominated by the frequency components present in the incident signal. As with the EFF the amplitudes of these components are slightly unequal.

In the second experiment the amplitudes in the incident signal are unequal (fig. 8) and the input signal was given by $0.7f_1+0.3f_3$. Unequal amplitudes were chosen to test whether the components in the gate array output also consisted of unequal components of the corresponding frequencies of approximately the same proportions.



(i)

(ii) (iii)

(iv) (v)

Figure 8: Incident signal $0.7f_1+0.3f_3$ (i), output and frequency responses for EFF (ii,iii) and FFF (iv,v)

In the third experiment a composite signal which consisted of the sum of two sine waves of amplitudes 0.5 and frequencies f_1 and f_{10} was presented to the gate arrays. This

signal is a much more severe test of the linear behaviour. This is because one frequency is in the passband (f_1) and the other well into the stop region (f_{10}). The results are shown in fig. 9.

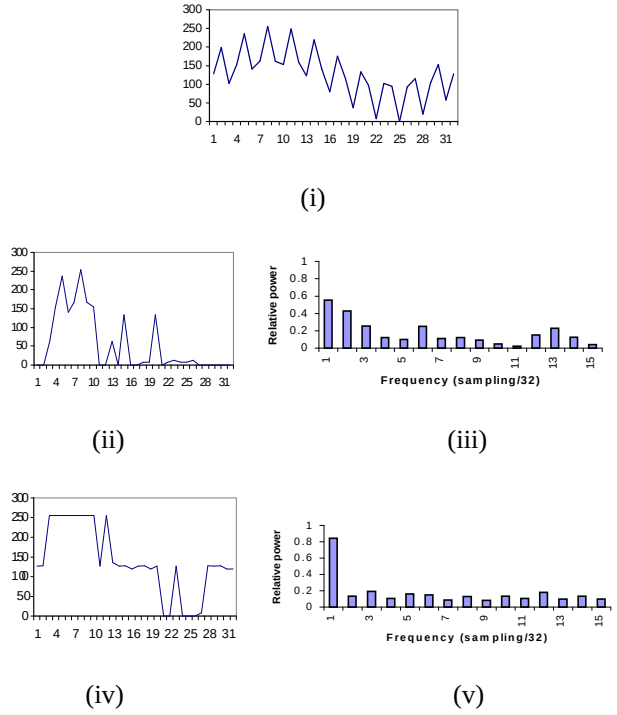


Figure 9: Incident signal $0.5(f_1 + f_{10})$ (i), output and frequency responses for EFF (ii,iii) and FFF (iv,v)

A perfect linear filter would filter out the high frequency component (f_{10}) and pass only an unattenuated sine wave of frequency f_1. The output response of the EFF evolved gate array is rather poor (9(ii)). Firstly the output signal shape is no longer very smooth and the second half of the cycle has virtually dissapeared. The frequency response (9(iii)) also shows the attenuation of the incident signal and the presence of significant off-frequency components. The EFF evolved gate array is responding in a much less linear way. The behaviour of the FFF evolved gate array is much better. Firstly the output response has many similarities to the ouput response seen in fig. 4 (iv), i.e. it is giving a 'squared' sinusoidal response. The frequency response (9(v)) confirms the considerably more linear behaviour as it is dominated by the fundamental with much less attenuation. Once again it should be emphasised that the gate arrays are responding to these composite signals for the first time. So the presence of this linearity is fairly remarkable.

4.3 Response to sums of three sine waves

As a severe test of the behaviour of the evolved gate arrays a signal was presented which comprised three sine components of unequal amplitudes. The dominant component had an amplitude of 0.5 and a frequency in the stop region (f_{10}), the other two components were f_1 with amplitude 0.2 and f_2 with amplitude 0.3. The incident signal is shown in fig 10(i) and the expected response of a perfect low pass filter is seen in fig 10(ii). The responses of the evolved gate arrays are seen in figs. 10(iii)-(vi).

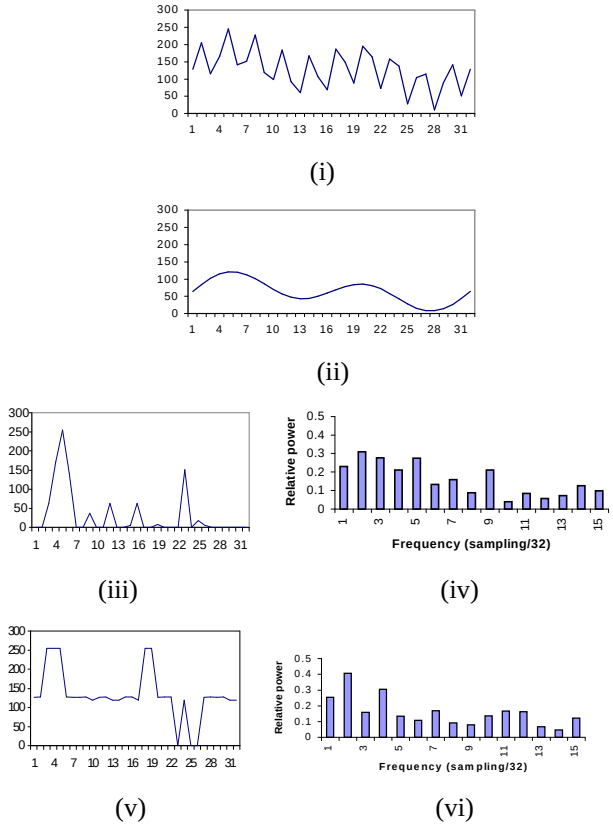
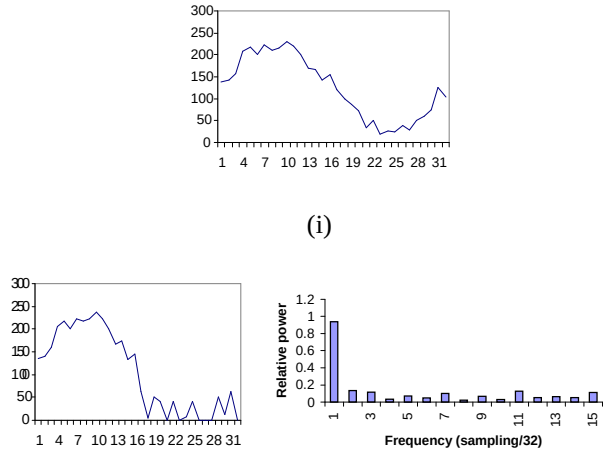


Figure 10: Incident signal $0.2f_1 + 0.3f_2 + 0.5f_{10}$ (i), output and frequency responses for EFF (iii, iv) and FFF (v,vi). The incident signal with stop frequencies removed $0.2f_1 + 0.3f_2$ is shown in Fig. 10 (ii)

It is clear that both gate arrays are not able to easily follow the low amplitude pass signal (10(ii)). The output signal shape for both gate arrays is poor. Even though the frequency domain is weakly dominated by low frequencies there are a large number of higher frequency components being transmitted.

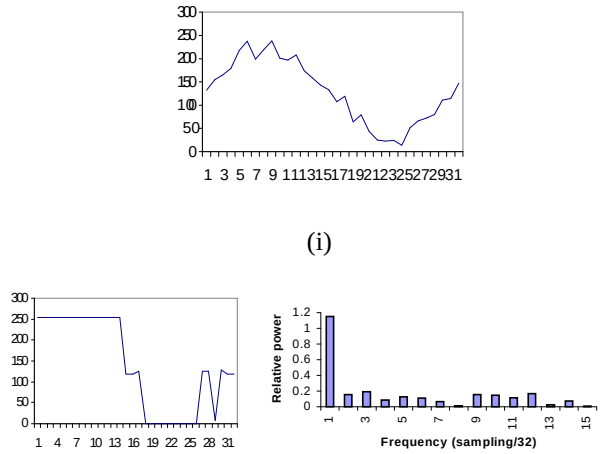
4.4 Noise tolerance

The behaviour of the evolved gate arrays to incident noisy signals was investigated. Figs. 11and 12 show the responses of the EFF and FFF gate arrays respectively when presented with a sine wave signal f_1 with an additional noisy component η which was defined as $\frac{1}{2}(1+\chi)2^{w}$, where χ is a uniform random real variable with mean 0, defined in the range [-1,1].



(i) (ii) (iii)

Figure 11: Incident signal $0.8f_1+0.2\eta$ (i), output (ii) and frequency response (iii) for EFF



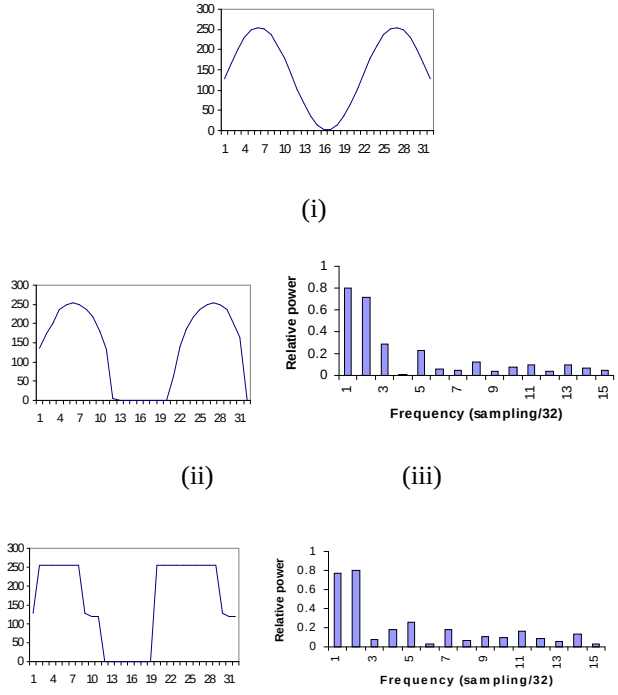
(i) (ii) (iii)

Figure 12: Incident signal $0.8f_1+0.2\eta$ (i), output (ii) and frequency response (iii) for FFF

Examining figs. 11(i) and (ii) it is clear that with the EFF the output response closely follows the noisy input signal in the positive going half-cycle but as was seen earlier (especially fig. 4) it doesn't respond well to low amplitude signals in the latter part of the sample window. The response of the gate array evolved with the FFF is interesting (fig. 12(ii)) as the added noise appears to have been removed. This fits in with the previous observation that the gate array evolved with the FFF appears to be operating in almost a three-level manner (0, 128, 255). The lack of sensitivity to noise could potentially be a useful property for real filtering tasks.

4.5 Non-integral frequency response

The final type of unseen signal presented to the gate arrays was a sine wave with non-integral frequency. The frequency of the incident signal was 1.5 times the fundamental. This causes it to have three half-cycles fitting in the sample window as shown in fig. 13(i). The output and frequency responses of the gate arrays are shown in the remaining figures.



(i) (ii) (iii) (iv) (v)

Figure 13: Incident signal $f_{1.5}$ (i), output and frequency response for EFF (ii,iii) and responses for FFF (iv,v)

Fig. 13(ii) is suprising as the EFF gate array response is very good and apparently *better* with this incident signal which was not in the training set than its response to individual signals which were part of the training set. It is also suprinsing in that it follows well the third positive – going half cycle. The output response of the FFF evolved gate array is also good but once again has lost the curvilinear shape of the incident signal. In the frequency domain (figs. 13(i) and (ii)) the power is shared in the neighbouring frequencies, this is a characteristic of the discrete Fourier transform.

5 Hardware aspects of evolved circuits and comparison with conventional

In this section the actual circuits corresponding to the best evolved gate arrays are examined. Firstly there are marked differences between the output connections of the EFF and FFF evolved gate arrays. The FFF evolved array has most of the eight physical outputs connected to the output of the same multiplexer. The output section of the chromosome is as follows: 94 , 94, 94, 79, 94, 94, 94, 71. The leftmost position is least significant. Thus the output bit streams look like *aaabaaac* where *a,b* and *c* are binary. This means that the FFF circuit is only capable of outputting eight numbers. These are as follows: 255, 247, 136, 128, 127, 119, 8, 0. Actually when the output data was inspected 247 and 8 didn't occur. It is now clear why the output responses of the FFF evolved gate array comprise approximately three levels: high (255), middle (136,128,127) and low (0). The output section of the chromosome corresponding to EFF evolved gate array is as follows: 90, 91, 90, 72, 91, 94, 92, 87 which can be written *abacbdef* where the letters represent independent binary bits. Thus there are 64 different levels of response.

Examining the way the evolved gate arrays make use of the input samples is also revealing. In the experiments reported in this paper a sample window of 6 was used with a wordlength of 8. Thus for each sample 48 bits grouped into 6 lots of 8 were presented to the gate array in order for it to produce an output. The evolved gate arrays tended to use only the most significant bits of each group of 8. Large numbers of input bits were completely ignored. The FFF gate array completely ignored the sixth sample in each sample window! Thus indicating that a filter order 6 was unnecessary. The EFF only used a single bit from this sample too. These findings suggest that one could obtain just as good filtering action using smaller wordlegths (perhaps as little as 4 bits).

The actual number of multiplexers used in the FFF gate array was 20 out of a possible 49 and the EFF gate array used 29. It is interesting to compare this with a conventional FIR filter implementing a finite difference equation. An eight bit multiplier alone would require 64 AND gates and 8x7=56 full adders. Each adder would require in its most efficient implementation 1 multiplexer and two exclusive-OR gates. To build this multiplier one would require 232 multiplexers. A 16 bit adder is also required to add up the six 16-bit products this would require 48 multiplexers. So one would require 280 multiplexers in total. A 16-bit register would also be needed to accumulate and store the sum during calculation. The evolved gate array 'filters' would operate much faster than a conventional design as the output is only limited by the speed of the gates, one doesn't have to wait for large multiplications, additions and accumulation cycles. One should however be very cautious about making such comparisons as clearly the gate array 'filters' are far from ideal at present.

6 Conclusions

It has shown in this paper how it is possible to evolve tiny gate arrays which exhibit digital filtering properties without using the concept of a linear difference equation. It was shown empirically that the gate arrays are often nearly linear in response. The gate arrays were evolved to have filtering properties by presenting digitised pure sine signals and assessing the output signals. Two fitness functions were defined. In one the fitness was calculated by looking at the responses of the gate arrays in the frequency domain by applying a dicrete fourier transform to the gate array output. The other fitness was calculated by examining the closeness of the gate array output to the the desired output. Experience suggests that gate arrays that are evolved using a fitness function which looks at the frequency spectrum of the circuit output appear to be more linear in behaviour than using an error based measure of fitness. However a considerable amount of further investigation is required to substantiate this observation. The circuits evolved with the Fourier fitness function are also input noise tolerant and can remove noise. Since the gate arrays are carrying out filtering without directly implementing a difference equation, there is no explicit multiplication and addition, and there are no coefficients! Currently there is no known mathematical way of designing filters directly at this level. The origin of the quasi-linearity is at present quite mysterious. There is much more work that needs to be done to understand the behaviour of such systems. A fundamental question that still remains is how the linearity of the filters may be enhanced. Naturally an important future step would be to actually build these gate arrays in hardware and directly investigate their ability to act as filters on sampled waveforms. This remains for the future.

References

[1] Arslan T., and Horrocks D. H., "A Genetic Algorithm for the Design of Finite Word Length Arbitrary Response Cascaded IIR Digital Filters", Proceedings of the First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA'95), No. 414, IEE, London, pp. 276-281, 1995.

[2] Bäck T., Hoffmeister F., and Schwefel H.-P., "A Survey of Evolutionary Strategies", in R. Belew and L. Booker (eds.), Proceedings of Fourth Int. Conf. On Genetic Algorithms, San Mateo, CA, Morgan Kaufmann, pp. 2-9, 1991.

[3] Beatriz Garmendia-Doval A., Mohan C. K., and Prasad M. K., "Evolving Tree Representations of Stack Filters", in J.R. Koza et al. (eds.), Genetic Programming: Proceedings of the 3rd Genetic Programming Conference, Morgan Kaufmann, San Francisco, CA, pp. 103-108, 1998.

[4] Chellapilla K., Fogel D. B., and Rao S. S., "Gaining Insight into Evolutionary Programming Through Landscape Visualization: An Investigation into IIR Filtering", Evolutionary Programming , pp. 407-417, 1997.

[5] Delibasis K. K., Undrill P. E., and Cameron G. G., "Genetic algorithm implementation of stack filter design for image restoration", ", IEE Proceedings in Vision, Image and Signal Processing, Vol. 143, No. 3, pp. 177-183, 1996.

[6] Dempster A. G., and Macleod M. D., "Use of Minimum-Adder Multiplier Blocks in FIR Digital Filters", IEEE Transactions in Circuits and Systems-II: Analog and Digital Signal Processing, Vol. 42, No. 9, pp. 569-577, 1995

[7] Esparcia Alcazar A. I., and Sharman K. C., "Some Applications of Genetic Programming in Digital Signal Processing", in Late Breaking Papers at Genetic Programming 96, Stanford, pp. 24-31, 1996.

[8] Harris S. P., and Ifeachor E. C., "Automating IIR filter design by genetic algorithm", Proceedings of the First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA'95), No. 414, IEE, London, pp. 271-275, 1995.

[9] Iba H., Iwata M., and Higuchi T., "Machine Learning Approach to Gate-Level Evolvable Hardware", in Higuchi T., Iwata M., and Liu W., (eds.), Proc. of The 1st International Conference on Evolvable Systems: From Biology to Hardware (ICES96), LNCS, Vol. 1259, Springer-Verlag, Heidelberg, pp. 327 – 343, 1997.

[10] Ifeachor E. C., and Jervis B. W., "Digital Signal Processing: A Practical Approach", Addison-Wesley, 1993

[11] Miller J. F., and Thomson P., "Evolving Digital Electronic Circuits for Real-Valued Function Generation using a Genetic Algorithm". Koza, John R. et al, (eds.). Genetic Programming: Proceedings of the 3rd Annual Conference, Madison, Wisconsin. San Francisco, CA: Morgan Kaufmann pp. 863-868, 1998.

[12] Miller J. F., Thomson P., "Aspects of Digital Evolution: Evolvability and Architecture", in Eiben A. et al (eds.), Proceedings of The 5th International Conference on Parallel Problem Solving from Nature (PPSNV), Vol. 1498, Springer-Verlag, Heidelberg, pp. 927-936, 1998.

[13] Miller J. F., Thomson P., "Aspects of Digital Evolution: Geometry and Learning", in Sipper M., Mange D., and Perez-Uribe A. (eds.), Proceedings of The 2nd International Conference on Evolvable Systems: From Biology to Hardware (ICES98), LNCS, Vol. 1478, Springer-Verlag, Heidelberg, pp. 25-35, 1998.

[14] Miller J. F., Thomson P., and Fogarty T. C., "Designing Electronic Circuits Using Evolutionary Algorithms. Arithmetic Circuits: A Case Study", in Genetic Algorithms and Evolution Strategies in Engineering and Computer Science: Quagliarella D., Periaux J., Poloni C. and Winter G. (eds.), Wiley, 1997.

[15] Miller J. F., "Evolution of Digital Filters using a Gate Array Model", Proceedings of the First EvoIASP'99 Workshop on Image Analysis and Signal Processing, Goteborg, Sweden, LNCS Vol. 1596, Springer-Verlag, Heidelberg, pp. 17-30, 1999.

[16] Miller J. F., "Digital FIR Filter Design at Gate-Level using Evolutionary Algorithms", GECCO'99, Orlando, USA,July, 1999.

[17] Murakawa M., Yoshizawa S., and Higuchi T., "Adaptive Equalisation of Digital Communication Channels Using Evolvable Hardware", in Higuchi T., Iwata M., and Liu W., (eds.), Proceedings of The 1st International Conference on Evolvable Systems: From Biology to Hardware (ICES96), LNCS, Vol. 1259, Springer-Verlag, Heidelberg, pp. 379 – 389, 1996.

[18] Redmill D. W., and Bull D. R., "Design of Low Complexity FIR Filters using Genetic Algorithms and Directed Graphs", in Proceedings of the Second IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA'97), No. 446, IEE, London, 1997.

[19] Sriranganathan S., Bull D. R., and Redmill D. W., "Design of 2-D Multiplierless FIR Filters using Genetic Algorithms", pp. 282-286, Proceedings of the First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA'95), No. 414, IEE, London, 1995..

[20] Sundaralingam S., and Sharman K. C., "Genetic Evolution of Adaptive Filters", in Proc. of DSP, London UK, pp. 47-53, 1997.

[21] Thompson A., "An evolved circuit, intrinsic in silicon, entwined with physics", in Higuchi T., Iwata M., and Liu W., (Editors), Proceedings of The 1st International Conference on Evolvable Systems: From Biology to Hardware (ICES96), LNCS, Vol. 1259, Springer-Verlag, Heidelberg, pp. 390 – 405, 1997.

[22] Thompson A., "On the Automatic Design of Robust Electronics Through Artificial Evolution", in in Sipper M., Mange D., and Perez-Uribe A. (eds.), Proceedings of 2nd International Conference on Evolvable Systems: From Biology to Hardware (ICES98), LNCS, Vol. 1478, Springer-Verlag, Heidelberg, pp. 13 -24, 1998.

[23] Wade G., Roberts A., and Williams G., "Multiplier-less FIR filter design using a genetic algorithm", IEE Proceedings in Vision, Image and Signal Processing, Vol. 141, No. 3, pp. 175-180, 1994