Aspects of Digital Evolution: Evolvability and Architecture

Julian F. Miller[1] and Peter Thomson[1]

[1]Department of Computing, Napier University, 219 Colinton Road, Edinburgh, EH14 1DJ, UK. Email: j.miller@dcs.napier.ac.uk, p.thomson@dcs.napier.ac.uk
Telephone: +44 (0)131 455 4305

Abstract. This paper describes experiments to determine how the architecture *vis-a-vis* routing and functional resources affect the ease with which combinational logic designs may be evolved on a field-programmable gate array (FPGA). We compare two chromosome representations with differing levels of connectivity, and show that the amount of routing and functional resource have a marked effect on the success of the evolutionary process.

0. Introduction

There is, at the current time, a growing interest in designing electronic circuits using evolutionary techniques [11]. Koza [8] showed how simple digital circuits could be evolved using Genetic Programming, and Iba et. al. [5] showed how it was possible to design circuits by evolving the functionality and connectivity of interconnected AND, OR, and NOT gates for intended use on a programmable logic array device (PLA). The group at EPFL developed a cellular automaton in hardware using Xilinx XC6216 Field Programmable Gate Arrays (FPGAs) which was able to carry out global synchronisation tasks despite the fact that the cells behaviour was determined locally [2]. Thompson [12] used a genetic algorithm to directly evolve configuring bit strings for the Xilinx 6216 chip to create a circuit which would carry out a frequency discrimination task. Other workers have evolved digital circuits with a variety of representations [3, 4, 7]

Our own early work evolving digital electronic circuits using genetic algorithms has concentrated on arithmetic circuits [1][10]. Although we showed that it was possible to evolve novel designs for small circuits (e.g. the two-bit multiplier) we recognised how much more difficult it became to evolve circuits with just a modest increase in function complexity, e.g. the three-bit multiplier.

In this paper our motivation is the understanding of the design factors which influence the ease with which we can evolve 100% functionally correct digital combinational circuits. We contend that the issue of the circuit *architecture* is of primary importance here. The circuit architecture means essentially how the digital logic is implemented using logical sub-blocks, how complex is the logic within the sub-blocks,

and how the sub-blocks may be connected together. The particular model of the architecture we are concerned with here is known as *fine-grained*. In this architecture the circuit is implemented using an array of simple cells, each of which may become any two-input logic gate or single control multiplexer. Also the cells have a regular connection pattern. Such an architecture is provided on the Xilinx XC6216 FPGA. Here we are concerned with the relative importance of the amount of functional cells and the connectivity (routing) of those cells in the *evolvability* of combinational logic circuits.

We have already devised two different chromosome representations A and B for feed-forward logic circuits [1][9][10]. Type A chromosome was a netlist - a set of interconnections and gate level functionality for a rectangular array of cells which form the connections between the primary inputs and outputs. The inputs of cells in a particular column could be connected to the outputs of cells in previous columns (or indeed the primary inputs) according to a *levels-back* connectivity parameter. The type B chromosome was again a netlist, however it simulated exactly the architecture of the Xilinx 6216 FPGA. Here we compare the performance of the two representations and then go on to examine in some detail how varying the connectivity and functionality of the array of cells affects the evolvability of a two-bit binary multiplier circuit.

1. Two Chromosome Representations of a cellular array

1.1 Chromosome A

Consider a 3 x 3 array of logic cells between two required primary inputs and two required outputs using a chromosome of type A. The inputs 0 and 1 represent fixed values, logic ‘0’ and logic ‘1’ respectively.

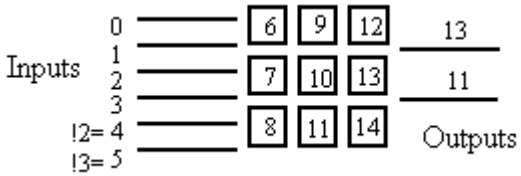


Fig. 1. A 3 x 3 geometry of uncommitted logic cells with inputs, outputs and netlist numbering.

The inputs (two in this case) are numbered 2 and 3, with 2 being the most significant. The lines 4 and 5 represent the inverted inputs 2 and 3 respectively. The logic cells which form the array are numbered column-wise from 6 to 14. The outputs are numbered 13 and 11, meaning that the most significant output is connected to the output of cell 13 and the least significant output is connected to the output of cell 11. These integer values, while denoting the physical location of each input, cell or output within the structure, now also represent connections or *routes* between the various points. Each of the logic cells is capable of assuming the functionality of any two-

input logic gate, or, alternatively a 2-1 *multiplexer* (MUX) with single control input. A sample chromosome is shown below:

0 2 -1 1 3 -5 2 4 3 2 6 7 0 8 -10 7 8 -4 6 11 9 6 4 -9 2 11 7 13 11

Fig. 2 A type A chromosome for the 3 x 3 geometry of Figure 1

In this arrangement the chromosome is split into groups of three integers. The first two values represent points to which the first and second inputs of the gate are connected. The third value may either be positive - in which case it is taken to represent the control input of a MUX - or negative - where it is taken to represent a two-input gate. The inputs are labeled A and B for convenience. The following notation is used: (i) & = AND, (ii) | = OR, (iii) ^ = exclusive-OR, and (iv) ! = NOT. For the chromosome, the allowed functions are listed in Table 1, where -1 indicates A&B, -2 indicates A&!B and so on through to -12.

1.2 Chromosome B

The Xilinx 6216 FPGA internal cell architecture is an arrangement of 64 x 64 simple cells. Each cell can be either a two-input logic gate or a 2-1 line MUX.
Since we are considering only combinational designs it is vital to allow only *feed-forward* circuits. To achieve this we chose a cell connection scheme in which inputs are fed into a cell which are *East-going* or *North-going* only. If the cell is a MUX then we allow the control input of the MUX to arrive from the North (indicated by 'S'). In Figure 3 the cells are numbered according to their column and row position with the origin at the bottom left hand corner of the array. All arrows pointing towards (outwards from) a cell represent inputs (outputs). The primary inputs connect to cells on the leftmost column and lowest row. The primary outputs exit the cells which are located on the topmost row and the rightmost column (in this case cells in column and row two).

The cells are allowed to be one of the types shown in Table 1. A and B represent 'E' and 'N' inputs and C represents 'S' input. If a cell at position (col, row) is a MUX then the 'S' input is assumed to the 'E' output of the cell located at position (col-1, row+1). If the MUX cell is located at column zero then we take the control input of the MUX from the primary input located at position (-1, row+1). In such a scheme cells in the top row are not allowed to be MUXs.

The chromosome has four parts; functional, routing, input, and output. The functional chromosome is a set of integers representing the possible cell types as indicated by gate type in Table 1.

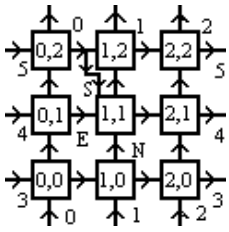


Fig. 3. Cell connections for Chromosome B

For N cells in the rectangular array there are N pairs (i,j), i ,j ε {0,1,2}, of routing genes which make up the routing chromosome, where the first (second) element of the pair represents the North (East) output. If i or j equals 0 then the North (East) output is connected to the East (North) input, and if i or j equals 1 then the North (East) output is connected to the North (East) input. If a routing gene is 2 then the corresponding cell output is a function of the cell inputs. Thus, the routing chromosome ignores the corresponding gene in the functional chromosome if its value is set to 0 or 1. The input chromosome has #rows + #columns elements. Each element can take any integer from 0 to #primary_inputs - 1. The output chromosome has #primary_outputs elements which represent the places in the cellular array from which the primary outputs are to be taken. To illustrate the interpretation of the chromosome consider the chromosome example (Table 2), which represents a chromosome for a 2x3 array of cells. For the sake of argument imagine that the target function is a 1-bit adder with carry. This has three inputs A, B, Cin and two outputs S and Cout.

Table 1. Allowed functionalities of cells

Gate Type	Function	Gate Type	Function
-4	!A & !C + !B & C	6	A ^ B
-3	A & !C + !B & C	7	A | B
-2	!A & !C + B & C	8	!A & !B
-1	A & !C + B & C	9	!A ^ B
1	0	10	!B
2	1	11	A | !B
3	A & B	12	!A
4	A & !B	13	!A | B
5	!A & B	14	!A | !B

We read the chromosome in the cell order (0,0) - (2,0) and (0,1) - (2,1) following from Figure 3. The inputs and outputs are also read in as shown in Figure 1. Thus the cell at (1,1) is a MUX of type - 1 (see Table 1). Its outputs as indicated by (2,2), in the routing part, are routed out to North and East. The inputs on the bottom row are Cin, A, A, and along the left edge, Cin and B. The Outputs S and Cout are connected

to the top row middle cell and bottom right hand corner cell (east output) respectively.

In the genetic algorithm we used uniform crossover with tournament selection (size 2). Winners of tournaments are accepted with a probability of 0.7. The routing chromosomes are all initialised to 2; 0 and 1 are only introduced by mutation, this is found to be more effective than random initialisation., this allows initially the circuit to use the maximum number of functional cells. Figure 4 demonstrates that this is advantageous. We use a fixed mutation rate which can be expressed as the percentage of all genes in the population to be randomly altered. The breeding rate represents the percentage of all chromosomes in the population which will be replaced by their offspring.

Table 2. Example chromosome for 2x3 array

Functional Part	Routing Part	Input part	Output part
2,9,11,12,-1,6	0,1,2,1,1,2,2,2,2,2,0,0	2,0,0,2,1	3,1

2. Functional and Routing Cell Resource Allocation

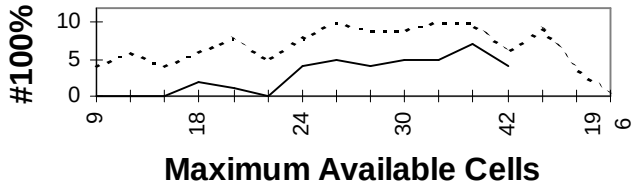
In considering new architectures for digital circuit evolution, there are two key issues: (a*) functionality of cells* - where evolution selects the logical functionality of a particular cell, and determines whether or not that cell should possess functionality, and (b) *routing* - where the routes that provide the interconnect between functional cells are evolved into non-functional cells (or functional cells which are partially used as routes).

The routeability of circuits is an important practical issue which is aptly demonstrated by a combinational design problem known as SBOX. This is a circuit used in data encryption. Traditional synthesis techniques are not able to produce a design which will place and route on to the Xilinx 6216 within the geometric bounding-box currently desired by engineers. This shows the importance of designing in such a way as to incorporate both issues of functionality of circuits *and* also the manner in which they route on target devices.

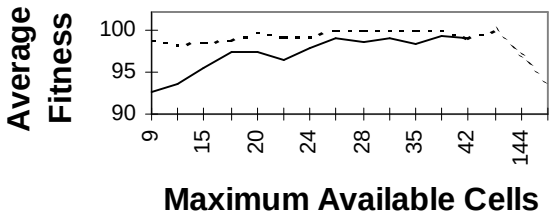
We conducted four sets of experiments. The first contrasted the relative effectiveness of evolution using type A and type B chromosomes. The second looked at the effectiveness of allocating functional and routing cells in particular patterns. The third compared the performance obtained using a differentiated cellular structure with that of an undifferentiated structure for a fixed maximum available functional resource. The fourth was to allocate increasingly more routing resource for a fixed average number of functional cells. The purpose being an assessment of the dependence of the evolvability of circuits with relative amounts of functionality and routing.

3. Results

In all experiments the average fitness was calculated over 10 iterations of the GA. The population size was 50, the breeding rate was 100% and the mutation rate 1%. Elitism was always used as it improves the performance markedly [9]. In the first set of experiments we compared the performance of chromosome A (levels-back = 2) with chromosome B. Figure 4 below shows the result of this. Figure 4(a) shows the numbers of instances of 100% functionally correct circuits for the two approaches, while Figure 4(b) shows the performance in terms of average fitness achieved.



(a)



(b)

Fig. 4. Comparison of Cell Structure A (broken) with Cell Structure B

Clearly, the chromosome representation A is able to evolve the circuit more easily (shown as a broken line). This is not entirely surprising because of the much greater freedom that A has in its choice of inter-cell routing. However, it does show that routing is important. Note that when the number of available cells reaches a value of around 40 the two approaches have approximately the same effectiveness. This may be because the amount of function and routing resource is so great that the second more constrained representation has become rich enough in resource to achieve a similar level of performance. One other noteworthy feature of Figure 4 is the almost constant level of performance of the representation A. This is surprising because the search-space is growing enormously over this interval. However, the solution space may be growing at the same rate. It may be that in these problems the *navigability* of the fitness landscape is more important than it overall size.

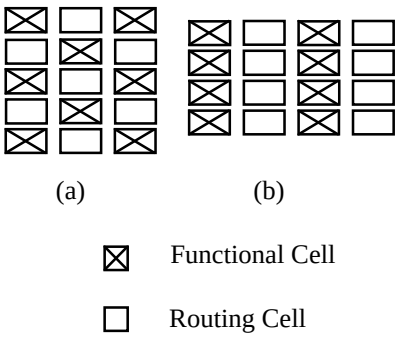


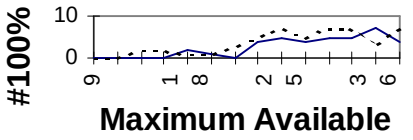
Fig. 5. The chequerboard and stacked pattern of functional cells

In the second set of experiments we deliberately *differentiated* functional and routing cells in particular patterns - chequerboard or stacked. This is shown in Figure 5.

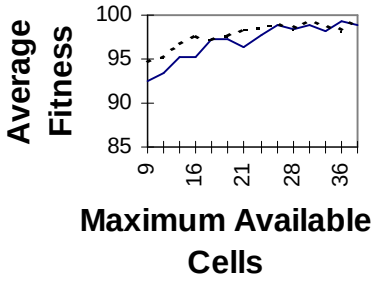
These patterns were preserved throughout the run using directed mutation. We found the two schemes made very little difference to the success of evolution - as measured by the average fitnesses that were obtainable. This is an interesting result because it shows that when the same proportion of routing resource is available to two different internal structures, they produce comparable results. Therefore, it may be that it is not so much the structure of functional and routing resources that is so important, but rather the ability for each functional cell to find many *alternative* routes to neighbouring cells. This freedom was the essential difference between approaches A and B. This conclusion does not mean that differentiation of cells is ineffective. Many more experiments need to be carried out to assess the actual way in which differentiation should take place.

In our third set of experiments we compared the effectiveness of a differentiated cell approach with an undifferentiated scheme. In the differentiated strategy, cells are forced to be either functional or routing - although some of the functional cells can through mutation and crossover incorporate internal routes. In order to compare a differentiated scheme with an undifferentiated one, we have to double the number of cells. This is so that the maximum number of available functional cell outputs was the same for the two methods. The results are shown in Figure 6.

Clearly, the differentiated approach (shown as a broken line) is the more effective of the two up to approximately 25 available functional cells. Thus it appears that a differentiated scheme eases the evolutionary process, and must be taken into account in the design of an architecture suitable for evolution. It is also interesting to note the very similar average fitness growth with increasing number of available cells. This suggests that the number of functional cells is the dominant factor. This is probably because there are many different ways in logic of building the same circuit functionality, and the increasing availability of cells which provide these options leads to a greater chance that the evolutionary path through the space of all possible circuits will be successful.



(a)



(b)

Fig. 6. Comparison of Differentiated and Undifferentiated Cell Structures

In the fourth set of experiments we allocated functional cells with different probabilities. A geometry of cells is selected, then functional cells are introduced with a chosen probability. This means that for a probability of, say, 0.1, and a geometry of 16 cells, then 3.2 cell outputs (each cell possesses 2 actual outputs) on average will be allocated as functional. The remainder become routing cells. Clearly, as the geometry is increased, the total amount of available routing resource is increased for the same probability. Therefore, we obtain a set of results which have similar numbers of functional cells but different amounts of routing. This enables us to compare how this eases the evolutionary process. The results of these experiments are detailed in Figure 7.

The graph of Figure 7 demonstrates that routing, whilst not dominant, is a vitally important consideration in the success of evolution as applied to circuit design. The graph shows that even when a structure has a more than adequate functional resource, it is still extremely difficult to evolve 100% correct designs if it is starved of routing options. The average fitness is unable to achieve the level that is mandatory (>96% on average) in order for 100% correct designs to emerge. This is very clear in the case where the number of cell outputs which are devoted to routing is between 0 and 5 (see routing allocation legend) on average. In this particular set of results, there are simply too few routing options for evolution to find any 100% correct solutions except perhaps where functional options dominate (at the higher end).

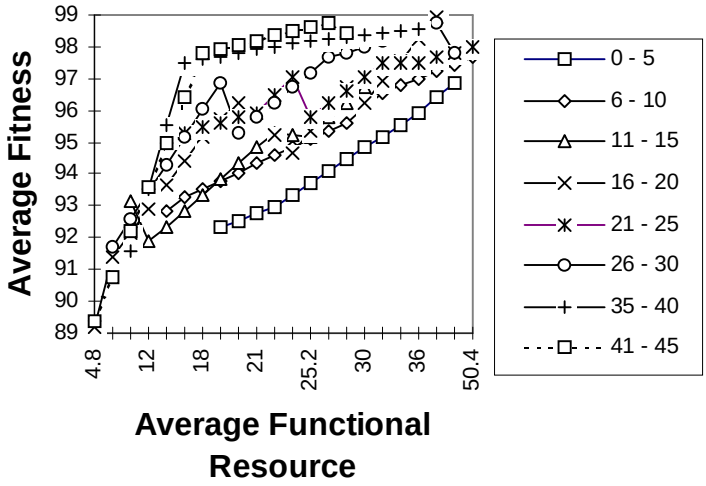


Fig. 7. Results for Functional against Fitness for Different Routing Resource

4. Conclusions

This paper has discussed the importance of the architecture to digital evolution. We show that the selection of the amount of functional and routing resource is important for the evolvability of circuits.

This is particularly highlighted in the case where the amount of functional resource is kept constant and the amount of routing resource is steadily increased. We saw that in the functionally starved domain (< 10 in Figure 7) the amount of routing resources provided is less important, whereas in the functionally enriched domain (>18), the availability of routing has a much more pronounced effect upon the success of evolution. The success of chromosome A relative to B leads us to conclude that one of the most important factors in designing an architecture specifically for evolution is the average number of neighbouring cells to which any particular cell may connect. We defined two differentiated cell structures and found a similarity of performance which may be explained by the fact that the average cell neighbourhood size was approximately the same. We intend to examine other differentiated schemes in which this neighbourhood size is varied more markedly.

Therefore, these two points should be taken into account when attempting to create any new architecture for evolution: (i) that adequate functional resources should be supplied, closely followed by adequate routing resources, and (ii) that these should be arranged in structures whereby full advantage can be taken of inter-function connectivity.

Another aspect which may affect the evolvability of circuits is the type of functional cell resource supplied (what gates are provided, in terms of type and complex-

ity). There is evidence that the choice of functionality within cells can also have a significant effect upon evolvability [6].

The Xilinx-based chromosome representation (B) that we currently use is very simple in structure and it may well be possible to develop more sophisticated representations - which will still be directly implementable on the device - but which take better advantage of neighbour connectivity. We intend to explore all of these issues in the near future.

References

[A] *Lecture Notes in Computer Science - Towards Evolvable Hardware*, Vol. 1062, Springer-Verlag, 1996.

[B] Higuchi T., Iwata M., and Liu W., (Editors), *Proceedings of The First International Conference on Evolvable Systems: From Biology to Hardware (ICES96)*, Lecture Notes in Computer Science, Vol. 1259, Springer-Verlag, Heidelberg, 1997.

1. Fogarty T. C., Miller J. F., and Thomson P.*: Evolving Digital Logic Circuits on Xilinx 6000 Family FPGAs*, in Soft Computing in Engineering Design and Manufacturing, P.K. Chawdhry,R. Roy and R.K.Pant (eds), Springer-Verlag, London, pages 299-305,1998.
2. Goeke M., Sipper M., Mange D., Stauffer A., Sanchez E., and Tomassini M., "Online Autonomous Evolware", in [B], pp. 96 -106
3. Higuchi T., Iwata M., Kajitani I., Iba H., Hirao Y., Furuya T., and Manderick B., "Evolvable Hardware and Its Applications to Pattern Recognition and Fault-Tolerant Systems", in [A], pp. 118-135.
4. Hemmi H., Mizoguchi J., and Shimonara K., " Development and Evolution of Hardware Behaviours", in [A], pp. 250 - 265.
5. Iba H., Iwata M., and Higuchi T., Machine Learning Approach to Gate-Level Evolvable Hardware, in [B], pp. 327 - 343.
6. Kalganova T., Miller J. F.: *Some Aspects of an Evolvable Hardware Approach for Multiple-Valued Combinational Circuit Design*, submitted to ICES'98, March 1998.
7. Kitano H., "Morphogenesis of Evolvable Systems", in [A], pp. 99-107.
8. Koza J. R., *Genetic Programming*, The MIT Press, Cambridge, Mass., 1992.
9. Miller J. F., Thomson, P.: *Aspects of Digital Evolution: Geometry and Learning*, submitted to ICES'98, March 1998.
10. 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: D. Quagliarella, J. Periaux, C. Poloni and G. Winter (eds), Wiley, 1997.
11. Sipper M., Sanchez E., Mange D., Tomassini M., Perez-Uribe A., and Stauffer A.: *A Phylogenetic, Ontogenetic, and Epigenetic View of Bio-Inspired Hardware Systems*, IEEE Transactions on Evolutionary Computation, Vol. 1, No 1., pp. 83-97.
12. Thompson A: *An Evolved Circuit, Intrinsic in Silicon, Entwined with Physics*, in [B], pp.390-405.