Hierarchical Strategies for Efficient Fault Recovery on the Reconfigurable PAnDA Device —Supplementary Material—

Martin A. Trefzer, *Senior Member, IEEE,* David M. R. Lawson, Simon J. Bale,
James A. Walker, *Senior Member, IEEE,* and Andy M. Tyrrell, *Senior Member, IEEE*

Appendix A
Fault Sources

The presence of faults in electronic circuits is inevitable due to the many variables and complex steps involved when designing and manufacturing them. Fault tolerance is a research area which is of continued interest since the presence of faults decreases an electronic system's reliability, which can lead to severe consequences particularly in safety-critical systems. But also in cases where user experience is at stake, this can mean significant economic impact on a manufacturer. Faults can occur throughout the entire design and fabrication cycle of an electronic system, during conceptualisation, specification, design, verification, fabrication and operation. This paper focuses on faults occurring during fabrication and at runtime, since at these stages a design is past the point where it can be permanently fixed and the power of reconfigurable architectures can really be harnessed to make such systems more reliable than those which cannot be reconfigured, i.e. the ones with fixed architectures. Hence, faults occurring as a consequence of manufacturing, operation and ageing are briefly discussed here. Based on this discussion and the fact that the experiments presented in this paper concentrate on correct circuit function, i.e. timing is not taken into account at this stage, a simple fault model regarding transistors as switches that can be "stuck open" (the transistor is always conducting irrespective of the gate voltage) or "stuck closed" (the transistor insulates and blocks any signal) has been adopted.

A.1 Manufacturing

Process technology for manufacturing semiconductor circuits has made tremendous progress when compared to the technology that was available 30 years ago, when reliably printing circuits in CMOS technologies as large as 3 micron posed major challenges. Transistor sizes have decreased alongside—or rather enabled by—advances in lithography and wafer processing, with devices now being as small as 14 nm. Even with careful control and calibration of process variables an inevitable facts of the physical world are noise and a degree of mismatch, which leads to variability in devices fabricated.

These variations are generally classed as systematic or stochastic (although there is no sharp divide), where systematic variations are a result of physical limitations to accuracy of manufacturing, e.g. mask alignment or ion beam energy over time, and stochastic variations are due to the atomic-scale size of the devices where the presence or absence of single atoms, due to inevitable random fluctuations during manufacture, start to matter and impact on device behaviour. These variations cause random variations in threshold voltages, speed and power, which may result in faults occurring due to timing violations or drive strength. As a consequence, it becomes more important, but also significantly higher effort, to create accurate simulation models and to accurately predict circuit behaviour. If statistically-enhanced device models are available, then the reliability of circuit simulations will be improved. However, the level of uncertainty in predicted circuit behaviour may still increase due to the feasibility of running a statistically sufficient number of Monte Carlo samples.

A.2 Environment

The environment in which a circuit is operated can have a significant effect on its performance and is therefore also a major cause for faults. The behaviour of electronic devices is temperature dependent, e.g. shifts in threshold voltage and carrier mobility affects circuit timing and may, again, cause faults due to timing violations. High temperatures increase power dissipation and circuit delay. This is caused by a number of phenomena that occur when temperature is increased: semiconductors' conductivity increases, which leads to larger transistor leakage currents, which further increase power dissipation leading to self-heating. In contrast, metals decrease in conductivity, increasing delays in circuits and dissipating additional heat. This leads to an increase of the thermal noise floor and introduces timing violations causing faults and errors. Prolonged exposure to heat can cause transistors to break down and become permanently conducting (on) or insulating (off).

Radiation is another environmental cause for circuit errors, malfunction or even permanent failure. Ionising radiation can induce currents and cause, in particular in

sequential circuits or memory cells, illegal states or wrongly altered memory contents leading to erroneous computation results. High-energy radiation can even alter the physical structure of devices through lattice displacement, which leads to a permanently altered behaviour or, in the worst case, to permanent failure of components.

A.3 Time

Transistor ageing refers to a number of phenomena which can cause devices to "wear out", which in the best case means a gradual shift towards worse performance and in the worst case device failure. Ageing is an inevitable cause for faults and errors. As transistor scaling has progressed, these effects have become more significant and current semiconductor technologies require both careful design and manufacturing to minimise stress, and therefore the impact of ageing, on the devices. Device ageing is an inevitable consequence of running a circuit over a period of time and environmental conditions are a major defining factor of the speed at which "wear out" takes place. There are a number of different mechanisms by which a transistor can age as described in more detail in [1] and [2]: bias temperature instability (BTI), which results from a charge build up in the dielectric over time, hot carrier injection (HCI), a stochastic process of high-energy electrons entering the gate dielectric, time-dependent dielectric breakdown (TDDB), resulting from charge trapping that ultimately can cause a short between source and drain, and electromigration, which is caused by moving electrons displacing material (metal tracks and contacts) over time breaking connections.

Appendix B
Synthesizing Logic Gates on PAnDA

Any binary function can be described by a boolean expression. For n variables there are 2^n possible combinations of binary values that those variables can take. Each one of these combinations can be described by an expression or minterm. By selecting all the minterms associated with the variable combinations for which a function should output a '1' and summing them together (with an OR operation), an expression for the logic function can be found. This initial mapping can often then be minimized in order to reduce the number of minterms and the number of variables in each minterm. This is commonly achieved with the Quine-McCluskey algorithm [3] in circuit design at the next abstraction layer up, building logic functions out of standard AND and OR gates.

An n-input binary logic function can be defined by a string of 2^n bits. Each bit of this string represents the function's output based on the input pattern equal to the bit's position in the string. Taking the 3-bit function in Table 1 as an example, the string defining it is 11011000 (the output column of the truthtable transposed). Table 1 shows the input states split into two groups, those that produce a 0 output and those that produce a 1. Here, these will be referred to as 0-states and 1-states respectively.

Reducing the number of required PAnDA slice branches works by merging branches with common factors. For example, consider two 1-states 000 and 001 that differ by only one input which means that the third input does not matter. If these two states were implemented as branches, changing the configuration of the PMOS CTs connected to the third input to be *disabled-conducting* would have no effect, they would now both conduct for both states, rather than one or the other. This means that one is now redundant and can be removed.

ABC	Y
000	1
001	1
010	0
011	1
100	1
101	0
110	0
111	0

0-States	1-States
010	000
101	001
110	011
111	100

TABLE 1
Left: truth table for the logic function used in the example. *Right:* the input states from the truth table split into two groups based on the output.

This process breaks down into two instances of boolean minimisation. If all the 1-states are written as minterms (e.g. $\overline{A}\overline{B}\overline{C}$, $\overline{A}\overline{B}C$, $\overline{A}BC$ ...), then a sum of products containing all of these represents the function required of the PMOS branches, i.e. whenever this sum of products is true, one or more PMOS branches must conduct. The same is true of the 0-states and the NMOS branches. Each product term can then be directly translated into a branch configuration using the following rules:

1) Any variables present in the product term represent *enabled* CTs.
2) Non-inverted variables map to *enabled* NMOS CTs and *enabled-inverted* PMOS CTs.
3) Inverted variables map to *enabled* PMOS CTs and *enabled-inverted* NMOS CTs.
4) Variables missing after minimisation map to *disabled-conducting* CTs.

Looking at the problem in this way enables the use of standard boolean minimisation algorithms such as Quine-McCluskey to minimise the expression and hence minimise the use of branches and CTs required. In addition to standard boolean functions, this method is capable of mapping tri-state and voltage divider circuits. For a tri-state function, the minterm(s) for any desired high-impedance states are not added to either the 0-states or 1-states lists. When the PMOS branch and NMOS branch functions are created, neither will conduct on the input(s) not included and so the output will be floating. Similarly, if desired, a minterm could be added to both lists and one of each type of branch could be made to conduct on that input. This is generally undesirable and could cause damage to the chip due to drawing large currents through the CTs.

Through minimising all 2^{16} boolean 4-input functions, it was found that all of them could be minimised into 8 product terms of each type or less, meaning that any 4-input boolean function can be implemented on a PAnDA-FÜNF slice. In addition, there are only two functions that don't minimise at all: odd and even 4-parity functions, which are inverses of each other.

APPENDIX C
STEP-BY-STEP ILLUSTRATION OF EXECUTING A STRATEGY LIST

In order to illustrate a typical result of the evolutionary algorithm, the first eight steps of an “unfixed”-optimised list for Z0 is worked through and explained in more detail here. Circuit Z0 is the inverter circuit with one PMOS and one NMOS CT on Input 0 of branches PMOS 0 and NMOS 0 (see Table 1). A visualisation of the first five steps is presented in Figure 1:

0) Strategy 2 - Inputs 0 and 3 are swapped on CAB 0. This is a sensible move as the two CTs used in the circuit are connected to input 0. If either had suffered just a conducting fault, this would recover from it.
1) Strategy 55 - NMOS 0 is swapped with NMOS 4. Again, this is sensible given the implemented circuit. The NMOS branch from the first CAB is moved to the 3rd CAB from the left. If the NMOS branch had suffered an insulating fault this would be a workaround. Though the NMOS CT had been moved to input 3 in the first strategy, CAB 2 will still be in the same state and so the Branch will be reordered to match CAB 2’s input order.
2) Strategy 17 - Inputs 2 and 3 are swapped on CAB 2. This has no effect in theory. Though an NMOS Branch has been moved to CAB 2, it is attached to Input 0 as CAB 2 has the default input order still. This shows potential for further improvement.
3) Strategy 5 - Inputs 2 and 3 are swapped on CAB 0. This moves the CT on PMOS Branch 0 to Input 2, potentially recovering from a conducting fault on input 3 of the PMOS 0 Branch.
4) Strategy 74 - NMOS 4 is swapped with NMOS 5. This certainly seems deliberate as the only NMOS CT used in the circuit is on NMOS 4.
5) Strategy 3 - Inputs 1 and 2 are swapped on CAB 0, moving the PMOS CT on PMOS 0 to Input 1. This CT has now been tried in all four positions on this branch.
6) Strategy 26 - PMOS 0 is swapped with PMOS 3. This make a lot of sense given that the previous strategy tried the active CT in the last possible position on PMOS 0. It was previously on Input B but will be translated to Input A when it moves to PMOS 3.
7) Strategy 6 - Inputs 0 and 1 are swapped in CAB 1. After moving the PMOS CT to a Branch in CAB 1 in the previous step, the Inputs are now swapped.

The selected strategy list for Z0 appears to have evolved in a useful way. Apart from the the third strategy, each change to the circuit puts the circuit into a state that has not been tried before. This is the obvious approach one would take to finding a working circuit on a faulty substrate where the location and nature of the fault is unknown. This result proves that lists of strategies for this purpose can indeed be generated automatically by means of an evolutionary algorithm.

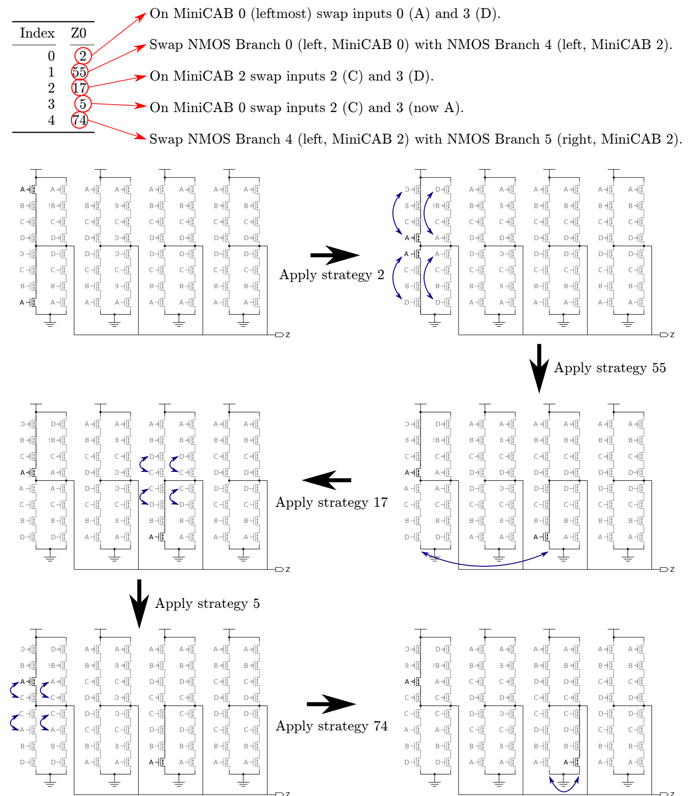

Fig. 1. A visualisation of the first five steps of a strategy list applied to circuit Z0 (an inverter). Note that apart from the third strategy applied (index 2), all the strategies transform an active part of the circuit, suggesting that the evolutionary approach to generating the list converges towards a useful solution.

APPENDIX D
STOCHASTIC STRATEGIES: NUMBER OF FAULTS TOLERATED FOR DIFFERENT BIASES

A separate experiment has been undertaken to measure how the different functions and biases perform under specific numbers of faults. For each of the functions Z0-Z3 10,000 experiments are carried out, continuously increasing the number of faults injected and using five different ratios between the two strategies used. Results are shown in Figure 2.

The findings echo those of the other experiments in that branch operations are the most effective strategies. Circuit Z0 is found to be the most capable of recovering from faults. This is because there are not many resources required for this circuit to work and so a large proportion of resources can be damaged whilst repair is still possible. Since only two transistors are required for this function to work, hence, as long as one branch of each type is free of insulating faults repair is possible. The effect of this can be seen directly from the results in Figure 2(a).

APPENDIX E
CIRCUIT DIAGRAMS OF THE FOUR TEST CIRCUITS

Circuit diagrams for all four test circuits are provided in Figure 3. The four test circuits have been selected with regards to their increasing resource requirements when mapped onto the PAnDA architecture.

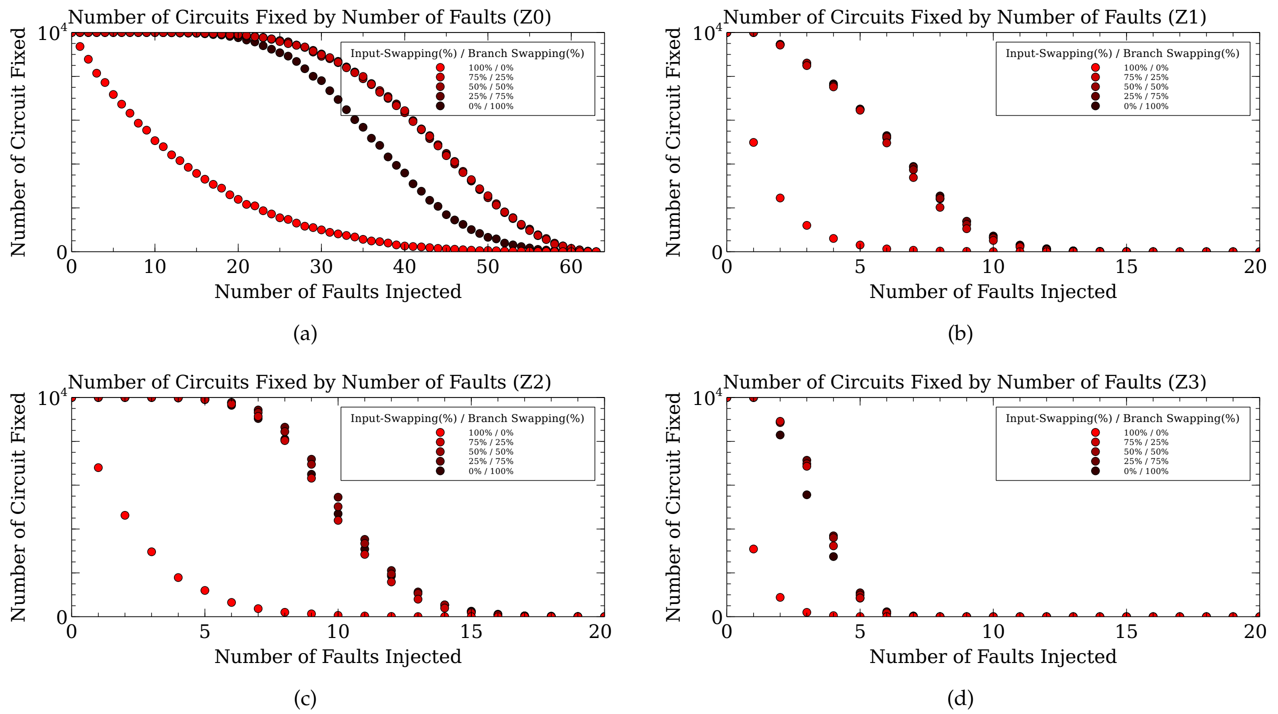

Fig. 2. The number of circuits fixed out of 10,000 samples when increasing numbers of random faults are introduced into circuit Z0 (*a*), circuit Z1 (*b*), circuit Z2 (*c*), circuit Z3 (*d*). Five biases of combining the two circuit transformations are tried for each number of faults. The results for circuits Z1, Z2 and Z3 were truncated to 20 faults as no more circuits were fixed after this point.

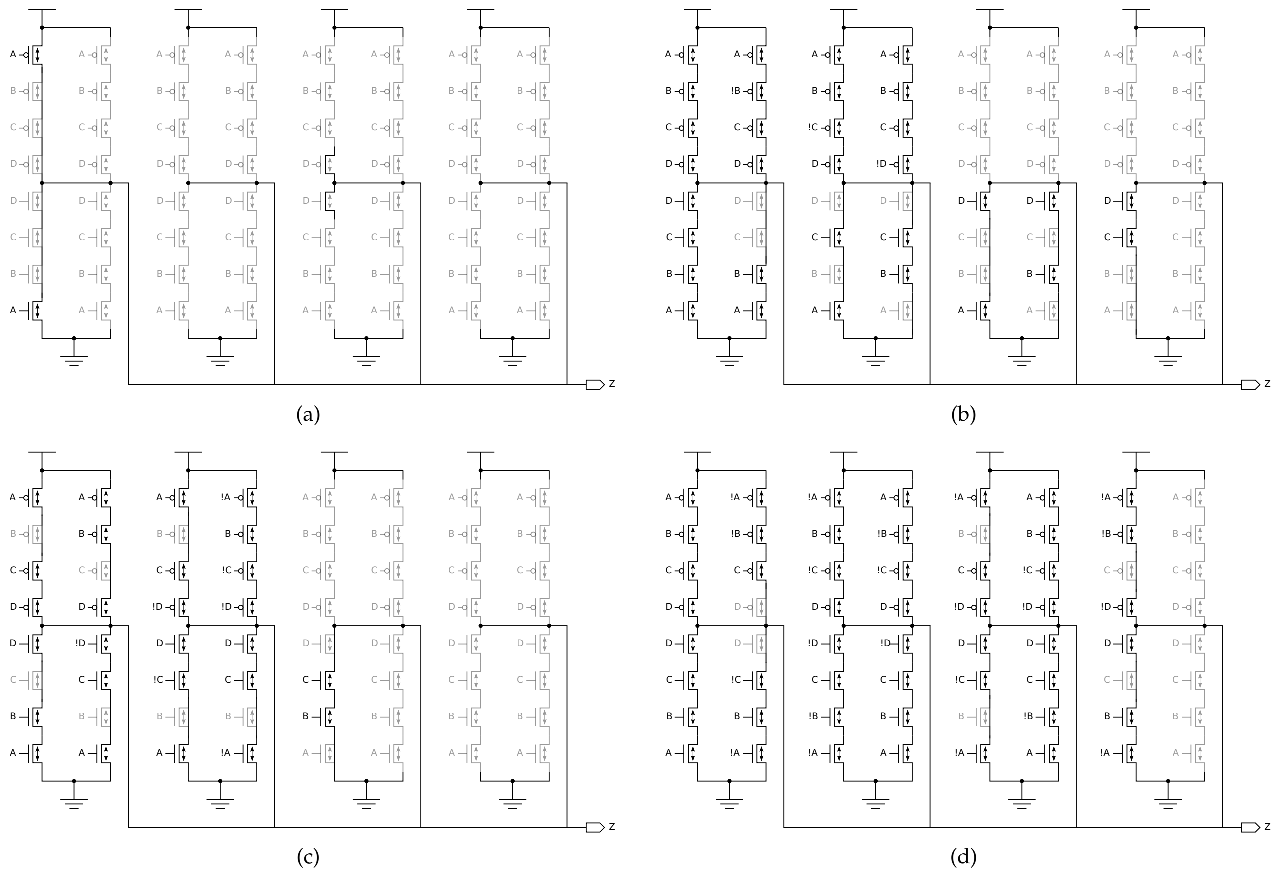

Fig. 3. Circuit diagrams for the test circuits Z0 (*a*), circuit Z1 (*b*), circuit Z2 (*c*) and circuit Z3 (*d*).

REFERENCES

[1] E. Maricau and G. Gielen, *Analog IC Reliability in Nanometer CMOS*. Springer Science + Business Media, 2013.
[2] J. Keane and C. H. Kim, "An odometer for CPUs," *IEEE Spectrum*, vol. 48, no. 5, pp. 26–31, May 2011.
[3] C. H. Roth Jr. and L. Kinney, *Fundamentals of Logic Design, 7th Edition*. Stamford: Cengage Learning, 2013.