Design of Single Electron Systems through Artificial Evolution

Adrian Thompson[*] and Christoph Wasshuber[†]

Abstract

We show how evolutionary methods can help in the design of single-electronic circuits with an example of evolving a simple NOR gate. Evolutionary algorithms, capturing the bare essentials of Darwinian evolution, work differently from conventional design methods, and have the potential to explore new territory. Our preliminary evolved circuit is far from an ideal NOR gate, but has interesting properties. It was evolved to work at a temperature of 340mK, and its performance deteriorates if the temperature is lowered, as well as if it is increased. This is contrary to the usual behaviour of single-electronic circuits, which generally improve with decreasing temperature. We hypothesise that the circuit exploits or relies upon the simulated effects of the particular thermal energies of the electrons at around 340mK.

Keywords: Nanoelectronic circuit design, evolutionary algorithms, design automation, single-electron circuit simulation, physics of computation.

1 Introduction

This paper discusses the use of evolutionary algorithms to design single electron systems, and describes a first exploratory experiment. Although evolutionary design does not circumvent all of the design challenges faced in making the technology viable, we show that some of its differences from conventional design can be helpful.

An evolutionary algorithm, while very different from evolution in nature, captures the bare essentials of Darwinian evolution: selection acting repeatedly upon heritable variation. Such algorithms [1] have been used in engineering design and optimisation since the 1960s [2], developed under the names of Evolution Strategies [2, 3], Evolutionary Programming [4], Genetic Algorithms [5] and Genetic Programming [6].

[*]Centre for Computational Neuroscience & Robotics, and Centre for the Study of Evolution, School of Cognitive & Computing Sciences, University of Sussex, UK. Email: adrianth@cogs.susx.ac.uk WWW: `http://www.cogs.susx.ac.uk/users/adrianth/`

[†]4106 Springhill Estates Drive, Parker, TX 75002, USA. Email: wasshub@parallelcomputers.com.

Applied to electronic circuit design, an evolutionary algorithm (EA) typically proceeds as follows. An initial population of one or more candidate circuit designs is created. These initial designs might be created entirely at random, or could be the product of a design or search procedure; perhaps they are even the results of a previous run of the evolutionary algorithm on some related or simpler problem [7]. Each individual candidate design in the population is then evaluated to give a single scalar measure of its performance, often referred to as 'fitness'.

The fitness evaluation procedure may take account of several performance criteria [8]. The primary criterion is usually the quality of behaviour, relative to a specification of the desired behaviour. The desired behaviour might be defined in terms of a target input⇒output relationship of the circuit, in the time or frequency domain. It is also possible to require a candidate circuit to *induce* a desired behaviour into a system in which it is embedded or to which it is coupled: for example a control circuit might be evaluated according to the behaviour of the robot it controls [9]. Non-behavioural criteria such as size, power consumption, fault-tolerance and testability can also be included in the fitness evaluation if they can be measured [10, 11, 12]. The candidate circuit's operation is often simulated in software for all these tests, but for some microelectronic circuits reconfigurable hardware such as Field-Programmable Gate Arrays (FPGAs) can be used [13, 14, 15].

Once all of the individual candidate designs in the population have been evaluated, some or all of them are selected to be 'parents'. One or more offspring are then created from these parents through the application of variation operators. The main variation operators are 'mutation', which introduces small stochastic changes, and 'crossover' (or recombination) which takes parts from two or more parents and brings them together into one offspring. The fitnesses of these new offspring designs are then evaluated, and some (or all) of the original population is replaced by some (or all) of the offspring. The procedure of this paragraph is repeated many times.

It is essential that the fitness values influence at least one of the three points of selection above: the selection of parents, the selection of individuals to be replaced by offspring, and the selection of offspring to be those replacements. If this is done so that the fitter individuals are more likely to survive to be parents, then the ingredients of Darwinian evolution are present. All being well, fitness values will increase over time until a circuit design adequate to the desired purpose is produced, at which point the process is stopped.

The evolutionary approach can only work when it is sufficiently likely that a pathway from the initial circuits to a satisfactory final one will be found through the successive application of the variation operators, guided through selection according to the fitness values. This depends not only on the fundamental structure of the design problem, but also on the fitness evaluation method, the selection method(s) and the variation operators. Note that if the operators are applied to an intermediate 'genetic' *representation* of the circuit design, then the way in which the actual design is encoded into this representation will influence the operators' effects. The search is seldom for a globally optimum design, but instead for any one of many possible adequate solutions.

Clearly, not all of electronics design can be tackled in this way. Since around 1995 there has been a considerable impetus of research to discern when, how, and why evolutionary techniques can be useful in electronics design [16, 17, 18, 19, 20, 21, 22]. In particular, a sequence of experiments at the University of Sussex, UK (summarised in [23, 24]) provides the basis for our evolution of single-electron designs to follow. These experiments were designed to determine how radically different evolved circuits can be from those produced by conventional design methods (see also [25]).

In practice, there are many important differences between evolutionary design and conventional methods. The most fundamental is that evolution proceeds through the accumulated action of the variation operators, and these can be largely *blind.* For example, a typical mutation operator considers each part of the design in turn, and with a small probability applies a random change to it. The consequences of the variation do not need to be predicted in advance; evolution works by taking account of the resulting changes induced into the measured performance of the system. Although it is possible to introduce context-sensitive or heuristic variation operators, it is this essentially stochastic and cumulative action of the variation operators that distinguishes evolution from other forms of heuristic search, and even from the iterative loop of design and testing often present in bottom-up electronics design.

Hence, circuits can be designed through evolution even when there is no feasible way of analysing how the individual interactions of the components give rise to the overall behaviour. This may be because the dynamics of the interactions are too complex, or simply because a tractable analytical model of the components is not available. For example, when one evolved microelectronic circuit was investigated [26], it was found that it had acutely exploited the semiconductor physics of the reconfigurable FPGA chip on which it was evaluated, even aspects that were not part of the chip's normal operation, and which were unknown to the investigators. By doing so, through ingeniously subtle means and a complex dynamics, the circuit was very much smaller than would normally be expected. This was possible because none of the conventional restrictive design rules were applied, since evolution does not have the same need for simplifying constraints. A decisive conclusion of the sequence of experiments [23, 24] was: EVOLUTION CAN EXPLORE BEYOND THE SCOPE OF CONVENTIONAL DESIGN.

Adaptive control algorithms (which could be evolutionary) have been proposed for the on-line tuning of nanoelectronic circuit parameters [27]. Evolutionary algorithms have also been applied to the characterisation of fabricated nanoelectronic devices [28], but not to the *design* of nanoelectronic systems. This paper now provides a case-study, as an initial exploration, and finishes by offering some conclusions. The experiments will be seen to hint at exciting prospects for the approach.

2 Case Study: Evolving a Single-Electron NOR Gate

We begin by investigating the evolution of a single-electron NOR gate because a simulator suitable for this purpose is available [29], a NOR gate is a simple circuit but a universal logic primitive, and single-electron NOR gates have been studied before [30]. Evolution of a small building-block to be used repeatedly in larger systems is attractive: The task is susceptible to contemporary evolutionary algorithms, which can be allowed to exploit subtly the physics of the medium at a fine level of detail, while leaving the higher-level logical composition of the building-blocks to more conventional methods.

The designs were represented as a two-dimensional array of nodes. The size was kept fixed at 7 rows $\times$ 4 columns. Between each pair of nodes was a component selected from the set {NONE, CAPACITOR, JUNCTION, WIRE}, where JUNCTION refers to a tunnel junction. Associated with each capacitor or junction was a real-valued capacitance in the range $\left[4.0 \times 10^{-19}, 1.0 \times 10^{-13}\right]$F. Also associated with each junction was a tunnel resistance in the range $\left[5.0 \times 10^{4}, 1.0 \times 10^{9}\right]\,\Omega$. NONE indicated the absence of a component between two nodes. A WIRE between two nodes was a virtual construct, signifying that the connected nodes should be amalgamated into one when the circuit is to be evaluated. Taking an example from the experiment to follow, Fig. 3 shows the representation of a circuit as manipulated by the evolutionary algorithm, while Fig. 4 shows the actual design so represented.

The top row of nodes was supplied with a constant bias voltage V_b in the range $\left[-1.0 \times 10^{-4}, -1.0 \times 10^{-6}\right]$V. The bottom row of nodes was connected to 0V. No components were allowed between the nodes in the top row, or between the nodes in the bottom row.

The two inputs to the NOR gate were always attached to the same nodes, as seen at the left in Fig. 3. Also shown is the fixed position of the 'preferred' output node. The actual output was taken from a valid output node closest in Manhattan distance to the preferred output node. A node was a valid output node if there was a connected path to it from each of the inputs that was at no point shorted to V_b or 0V. In the case of multiple valid output nodes of equal minimum distance from the preferred position, the one furthest to the right (and furthest down if there was still a tie) was chosen.

Although the array of nodes was fixed in size, the use of NONE and WIRE components, together with the output position selection method, gave considerable freedom for circuits to be of different sizes within the boundaries.

Underlying the choice of circuit representation was the notion that elements that interact should be adjacent to each other, since the usual action of a 'wire' is difficult to achieve [31]. To this end, regular cellular architectures have been proposed, e.g. [32]. Within a primitive (which may be repeated in some sort of array), the representation scheme adopted here is more flexible than a regular array, yet interacting elements can still be brought next to each other through a topology-preserving deformation of the layout once nodes connected by a virtual

WIRE have been amalgamated.

The fitness of a candidate circuit was evaluated by applying the voltage waveforms shown in Fig. 1 to the inputs, while monitoring the voltage at the output node. The input voltage sources were coupled to the circuit through fixed series capacitors of 3.333×10^{-13}F in a small attempt to model the situation where the inputs would be driven by the outputs of similar adjacent NOR gates; the output node was loaded by a 1pF capacitor to ground. These arrangements can be seen in Figs. 3 and 4.

Also shown in Fig. 1 is the ideal (target) output voltage waveform $V_{ideal}(t)$. If the actual observed output of a candidate circuit was $V_{out}(t)$, then its fitness over the trial of length T was calculated as:

$$\text{Fitness} = \frac{-1}{T\left|V_{true} - V_{false}\right|} \int_0^T \left|V_{out}(t) - V_{ideal}(t)\right| dt \tag{1}$$

or -1.0×10^6 if the individual was not viable. An individual was unviable if no valid output node could be found, or if an input was shorted to 0V or V_b. The fitness of a viable circuit, given by the equation, is just $-1\times$ the normalised mean error of the output voltage over the trial, so the perfect circuit would score 0.0.

The voltages V_{true} and V_{false} defining the logic levels of both the inputs and the ideal output could be varied by evolution (along with V_b) as part of the description of an individual circuit. They could assume any values, subject to the constraint that $|V_{true} - V_{false}| \geq 0.75V_b$, and they were initially randomly chosen from the interval $[V_b, 0.0]$V (again subject to the constraint). The input waveforms have nonzero rise and fall times, whereas the ideal output responds instantly at a logic threshold of $(V_{true} + V_{false})/2$. This provides a selection pressure for the evolution of noise margins.

The circuits were simulated using the SIMON package [33, 34, 29], with the parameter settings given in the footnote.[1] The first phases of the experiment were conducted at absolute zero temperature (0K), and neglected co-tunnelling (only first order tunnelling was simulated). In Phase 2 we will go on to describe how the temperature can be increased and higher-order tunnelling accounted for.

2.1 Experiment Phase 0

To start the experiment, an absolutely random search was conducted to find a prototype design with which to seed evolution. Each random individual was generated by selecting the component between each pair of nodes uniformly at

[1]All simulations used the quasi-stationary monte-carlo mode. For evolutionary fitness evaluations, where speed is important, a somewhat noisy simulation was used, with 3000 events simulated per timestep of 5ns. For Figs. 5, 6, 9, 10, and 11, higher resolution tests were performed, using 10000 events per 0.5ns timestep. The nodes always had zero initial charge, and a different random-number seed was supplied to the simulator for each fitness trial.

random from {NONE, CAPACITOR, JUNCTION, WIRE}, then choosing the associated capacitances, resistances, V_b, V_{true} and V_{false} uniformly at random from their permitted ranges. 58470 random circuits were evaluated, and the best one chosen. At this very early stage, the fitness values could be dominated by worthless transient charging/discharging, so the evaluations were preceded by the simulation of an initial settling time of 0.3125μs. During the settling time, the input voltages were both at V_{false} and the output was irrelevant. This settling time was not necessary during the evolutionary phases to follow.

2.2 Experiment Phase 1

The evolutionary algorithm was a fairly standard generational Genetic Algorithm (GA) [35]. A fixed-size population of 30 candidate circuit designs was maintained. Once their fitnesses had been evaluated, all except the single fittest member were replaced by offspring (a 'generation'). Individuals were selected (with replacement) to parent offspring with a probability proportional to the rank order of their fitness within the population, such that the fittest member's expected number of offspring was 2.0, the median-fittest's expected number was 1.0, and the least fit would get none. Baker's stochastic universal selection method [36] was used.

For each offspring individual, there was a probability of 0.7 that it would be generated through a crossover operation on two parents. Crossover was done by selecting uniformly at random a pair of rows and a pair of columns in the array of nodes. The rectangular region circumscribed by these limits would be taken from one parent, and the remainder from the other. If crossover was not performed (probability 0.3), then just one parent was taken.

To finish the formation of an offspring, it would be mutated as follows. Taking each component in turn, with a small probability it would be altered to a different uniformly randomly chosen component type. This probability was set such that the expected number of component-type mutations per individual was 0.7. Then, considering each component position in turn, with a small probability the capacitance value associated with that position would be perturbed (whether or not the component at that position happened to be currently of a type that used the capacitance value). The same perturbation procedure was then repeated for the resistance values. The probabilities of perturbations were set such that the expected number of capacitances altered per individual was 2.8, and the expected number of resistances altered was also 2.8. Finally, the global voltages associated with the circuit (V_b, V_{false} and V_{true}) were taken in turn and with a probability of 0.1 each, were perturbed.

The perturbations to the real-valued parameters were performed using the 'Breeder-GA (BGA)' mutation operator [37]. This operator takes the range of the variable as a parameter, and generates small perturbations with a higher probability than large ones. If, after mutation, $|V_{true} - V_{false}|$ broke the separation constraint, the two voltages were symmetrically moved further apart until they were valid. In the case of resistances and capacitances, which both range over many orders of magnitude, the log of the variable (and its range) was taken, the BGA mutation applied, and then the antilog of the result became the variable's new value.

Starting from a population of identical copies of the best circuit found through random search (Phase 0), the GA's operation consisted of many iterations of the generational cycle of fitness evaluations, selection of parents, offspring formation, and replacement of all but the currently fittest individual by the new offspring. Fig. 2 shows how the fitness rapidly increased at first, and then was slowly fine-tuned over a much longer period.

The GA was stopped after 8000 generations, and the best individual at that time is shown in Figs. 3 and 4. The behaviour shown in Fig. 5 is very close to the ideal, but it is also seen to be completely destroyed if we now enable the simulation of higher-order tunnelling events. If the temperature is increased (Fig. 6), the behaviour is completely lost at only 30mK.

In a similar earlier experiment at zero temperature [38] it was found that if evolution was continued from this point, but now with second-order tunnelling enabled, it took many generations to regain the original fitness. If third-order tunnelling was then enabled in the simulation, then again much more evolution was needed. Given that simulation of higher order tunnelling is extremely computationally expensive, the approach of gradually increasing the order of simulated tunnelling events, at zero temperature, does not appear fruitful. Attempts to conduct the experiment at a temperature of 500mK, even simulating only first order tunnelling, met with total failure: no initial circuits could be found (either through random search, or by the GA) that were above baseline fitness.

The way forward is suggested by noticing that the loss of behaviour with increasing temperature seen in Fig. 6 is gradual, although rapid.

2.3 Phase 2

Taking the final population of Phase 1, evolution was continued, still with only first-order tunnelling simulated. Whenever the fitness of the best individual reached a threshold of -0.25, the temperature was increased by 10mK. It can be seen in Fig. 7 that although these small temperature increases usually caused some loss in fitness, the population had enough residual performance for the evolutionary process to work on, in adapting the individuals to the new conditions.

Due to time constraints, the temperature was held constant once it reached 340mK, to allow a recognisable NOR-gate to be formed. When the experiment was terminated, the best circuit (Fig. 8) certainly would not work in a computational circuit, but can be seen to be roughly approximating the target NOR response.

The circuit has some interesting properties. Its response deteriorates only slightly if second-order tunnelling events are now included in the simulation, and there is no further degradation if third-order events are also modelled. Cotunnelling and increased temperature both smear out the Coulomb blockade in a very similar way; higher-order tunnel events are thus lumped into a larger effective temperature.

The thermal response of the circuit, considering only first-order tunnelling, is fascinating. Fig. 10 shows that the behaviour deteriorates not only when the temperature is increased, but also when it is *decreased*. The best performance is seen at 340mK — the temperature during the final stage of evolution. The simulation does not model thermal drift of the parameter values, so this curve means that the circuit exploits or relies upon the simulated effects of the particular thermal energies of the electrons at around 340mK. Further work would be needed to verify that these observations are physically realistic: perhaps this unusual circuit exercises the simulator outside of its normal validated domain of operation.

Although we have not produced an ideal NOR gate, its thermal response indicates that evolution has been exploring the utilisation of the (simulated) physical medium in ways not normally imagined.

3 Conclusion

We argued that evolutionary design is different from the usual processes in which human designers or conventional CAD tools engage. One consequence is that it is possible to evolve designs that take unusual leverage from the physics of their medium of implementation. This can be done even if there is no tractable analytical model to predict how the overall behaviour will emerge from the interactions of the components. At least for small systems, it can also be done with little prior conception of what kinds of design might be appropriate or effective. These properties are alluring for contemporary single-electronics, and the experiments were encouraging.

A representation scheme was developed that allows more flexibility than a regular array, yet maintains adjacency of interacting components. Some freedom was available to explore circuits of different sizes, and with their output in different positions. The bias voltage and the signal levels could co-evolve with the circuit structure. A method was found for the evolution of circuits to perform at nonzero temperature, which also appears to lessen the impact of co-tunnelling events, which we would rather not have to simulate during evolution. The resulting circuit does not serve as a good NOR gate, but does exhibit the ability of evolutionary techniques to navigate into intriguing unchartered territories of design.

Some important issues were not part of this first study. Perhaps the possibility of signal representation schemes other than voltage levels would be fruitful (e.g. [39]), and the issues surrounding the composition of evolved primitives into larger systems are worth closer attention. We ignored the effects of background charge and component tolerances, drift, and control, although parallel work has addressed the challenge of robustness in evolutionary microelectronics design [24].

It may be that a better NOR gate was not obtained because the fitness evaluations were quite noisy (visible in Fig. 7). By increasing the resolution of the simulation[2], and using a slightly different fitness function that penalises larger errors more heavily, a circuit with the improved response shown in Fig. 11 was obtained after a further 375 generations. When this circuit is simulated with multiple tunnelling events enabled, however, the response is no better than that seen in the main experiment (Fig. 9). Hence, although we might speculate that multiple tunnelling events may have less of an effect as the circuit becomes more closely adapted to the target response, or that evolution at temperatures higher than 340mK might also reduce this problem, it remains to be seen what experimental conditions are necessary for the evolution of a really practical NOR gate.

Most of the future directions mentioned above imply the use of more computationally expensive fitness evaluations. The set of experiments reported here (phases 0–2) took about 3 weeks on a dual-processor 466MHz PC. The computational demands are not necessarily terminal, as evolutionary algorithms parallelize very well to loosely-coupled MIMD parallel machines, such as cost-effective Beowulf-style clusters [40].

Nanoelectronics design seeks to employ subtle physics to do useful work. Natural evolution has done this in biology, and so can evolutionary algorithms in artificial media. Our experiments tentatively suggest that evolutionary methods may be a useful exploratory tool into novel kinds of design that may help to make such new technologies viable.

Acknowledgements

Thompson's work was funded under EPSRC grant GR/L60531. Special thanks to Phil Husbands for supporting this work within the Sussex Centre for Computational Neuroscience & Robotics.

References

[1] T. Bäck, D. Fogel, and Z. Michalewicz, editors. *Handbook of Evolutionary Computation.* Institute of Physics Publishing, 1997.

[2] I. Rechenberg. Cybernetic solution path of an experimental problem. Royal Aircraft Establishment, Library Translation 1122, 1965. Reprinted in 'Evolutionary Computation — The fossil record', D. B. Fogel, ed., chap. 8, pp297-309, IEEE Press 1998.

[3] H.-P. Schwefel and G. Rudolph. Contemporary evolution strategies. In F. Morán, A. Moreno, J. J. Merelo, and P. Chacon, editors, *Advances in Artificial Life: Proc. 3rd Eur. Conf. on Artificial Life*, volume 929 of *LNAI*, pages 893–907. Springer-Verlag, 1995.

[2] 10000 events were now simulated per 1ns timestep during evolutionary trials.

[4] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence Through Simulated Evolution.* John Wiley & Sons, Inc., 1966.

[5] J. H. Holland. *Adaptation in Natural and Artificial Systems.* Ann Arbor: University of Michigan Press, 1975.

[6] J R Koza. *Genetic Programming: On the programming of computers by means of natural selection.* MIT Press, Cambridge, Mass., 1992.

[7] P. Husbands, I. Harvey, D. Cliff, and G. Miller. Artificial evolution: A new path for artificial intelligence? *Brain and Cognition*, 34:130–159, 1997.

[8] Carlos M. Fonseca and Peter J. Fleming. An overview of evolutionary algorithms in multiobjective optimisation. *Evolutionary Computation*, 3(1):1–16, 1995.

[9] A. Thompson. Evolving electronic robot controllers that exploit hardware resources. In F. Morán, A. Moreno, J. J. Merelo, and P. Chacon, editors, *Advances in Artificial Life: Proc. 3rd Eur. Conf. on Artificial Life (ECAL95)*, volume 929 of *LNAI*, pages 640–656. Springer-Verlag, 1995.

[10] T. Kalganova and J. Miller. Evolving more efficient digital circuits by allowing circuit layout evolution and multi-objective fitness. In A. Stoica, D. Keymeulen, and J. Lohn, editors, *Proc. 1st NASA/DoD Workshop on Evolvable Hardware*, pages 54–63. IEEE Computer Society, 1999.

[11] R.S. Zebulum, M.A. Pacheco, and M. Vellasco. A multi-objective optimisation methodology applied to the synthesis of low-power operational amplifiers. In I.J. Chueiri and C.A. dos Reis Filho, editors, *Proc. XIII Int. Conf. on Microelectronics and Packaging*, volume I, pages 264–271. The Brazilian Microelectronics Society, 1998.

[12] Adrian Thompson. Evolving inherently fault-tolerant systems. *Proc Instn Mechanical Engrs, Part I*, 211:365–371, 1997.

[13] A. Thompson. Silicon evolution. In J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, editors, *Genetic Programming 1996: Proc. 1st Annual Conf. (GP96)*, pages 444–452. Cambridge, MA: MIT Press, 1996.

[14] Paul Layzell. Reducing hardware evolution's dependency on FPGAs. In *Proc. 7th Int. Conf. on microelectronics for neural, fuzzy and bio-inspired systems (MicroNeuro'99)*, pages 171–178. IEEE Comp. Soc. Press, 1999.

[15] A. Stoica, C. Salazar-Lazaro, D. Keymeulen, and K. Hayworth. Evolution of CMOS circuits in simulations and directly in hardware on a programmable chip. In W. Banzhaf, J. Daida, A. E. Eiben, et al., editors, *Proc. Genetic and Evolutionary Computation conference (GECCO-99)*, pages 1198–1203. Morgan Kaufmann, 1999.

[16] E. Sanchez and M. Tomassini, editors. *Towards Evolvable Hardware: The evolutionary engineering approach*, volume 1062 of *LNCS*. Springer-Verlag, 1996.

[17] T. Higuchi, M. Iwata, and L. Weixin, editors. *Proc. 1st Int. Conf. on Evolvable Systems: From Biology to Hardware*, volume 1259 of *LNCS*. Springer-Verlag, 1997.

[18] M. Sipper, D. Mange, and A. Pérez-Uribe, editors. *Proc. 2nd Int. Conf. on Evolvable Systems (ICES98)*, volume 1478 of *LNCS*. Springer-Verlag, 1998.

[19] J. Miller, A. Thompson, P. Thomson, and T. Fogarty, editors. *Proc. 3rd Int. Conf. on Evolvable Systems (ICES2000): From Biology to Hardware*, volume 1801 of *LNCS*. Springer-Verlag, 2000.

[20] X. Yao (Guest Editor). Special section on evolvable hardware. *Communications of the ACM*, 42(4):47–79, April 1999.

[21] M. Sipper and D. Mange (Guest Editors). Special issue on biology to hardware and back. *IEEE Trans. Evolutionary Computation*, 3(3), September 1999.

[22] A. Stoica, D. Keymeulen, and J. Lohn, editors. *Proc. 1st NASA/DoD workshop on Evolvable Hardware*. IEEE Computer Society, 1999.

[23] A. Thompson, P. Layzell, and R. S. Zebulum. Explorations in design space: Unconventional electronics design through artificial evolution. *IEEE Trans. Evol. Comp.*, 3(3):167–196, 1999.

[24] A. Thompson and P. Layzell. Evolution of robustness in an electronics design. In J. Miller, A. Thompson, P. Thomson, and T. Fogarty, editors, *Proc. 3rd Int. Conf. on Evolvable Systems (ICES2000): From biology to hardware*, volume 1801 of *LNCS*, pages 218–228. Springer-Verlag, 2000.

[25] J. R. Koza, F. H. Bennett III, M. A. Keane, et al. Searching for the impossible using genetic programming. In W. Banzhaf, J. Daida, A. E. Eiben, et al., editors, *Proc. Genetic and Evolutionary Computation conference (GECCO-99)*, pages 1083–1091. Morgan Kaufmann, 1999.

[26] Adrian Thompson and Paul Layzell. Analysis of unconventional evolved electronics. *Communications of the ACM*, 42(4):71–79, April 1999.

[27] R. W. Rendell and M. G. Ancona. Adaptive computation by interacting quantum dots. *Superlattices and microstructures*, 20(4):479–491, 1996.

[28] G. Klimeck, C. H. Salazar-Lazaro, A. Stoica, and T. Cwik. “genetically engineered” nanoelectronics. In A. Stoica, D. Keymeulen, and J. Lohn, editors, *Proc. 1st NASA/DoD Workshop on Evolvable Hardware*, pages 247–248. IEEE Computer Society, 1999.

[29] C. Wasshuber, H. Kosina, and S. Selberherr. SIMON – a simulator for single-electron tunnel devices and circuits. *IEEE Transactions on Computer-Aided Design*, 16:937–944, September 1997.

[30] R.H. Chen, A.N. Korotkov, and K.K. Likharev. Single-electron transistor logic. *Appl. Phys. Lett.*, 68(14):1954–1956, 1996.

[31] A. N. Korotkov, R. H. Chen, and K. K. Likharev. Possible performance of capacitively-coupled single-electron transistors in digital circuits. *Journal of Applied Physics*, 78(4):2520–2530, 1995.

[32] M. G. Ancona. Design of computationally useful single-electron digital circuits. *J. Appl. Phys*, 79(1):526–539, 1996.

[33] The Single-Electron Repository. http://home1.gte.net/kittypaw/.

[34] C. Wasshuber. *About Single-Electron Devices and Circuits.* PhD thesis, Technical University of Wien, Österreichischer Kunst- und Kulturverlag, 1997.

[35] D. E. Goldberg. *Genetic Algorithms in Search, Optimization & Machine Learning.* Addison Wesley, 1989.

[36] J. E. Baker. Reducing bias and inefficiency in the selection algorithm. In J. J. Grefenstette, editor, *Genetic Algorithms and their Applications: Proc. 2nd Int. Conf. on Genetic Algorithms (ICGA)*, pages 14–21. Lawrence Erlbaum Associates, 1987.

[37] H. Mühlenbein and D. Schlierkamp-Voosen. Predictive models for the breeder genetic algorithm. *Evolutionary Computation*, 1(1):25–49, 1993.

[38] A. Thompson. Evolutionary design for novel technologies. In *IEE Colloquium on Evolutionary Hardware Systems*, page 4/1, Savoy Place, London, June 1999.

[39] R. W. Rendell. Adaptive control of single-electron circuit signatures for computation. In M. Cahay, D. J. Lockwood, J.-P. Leburton, and S. Bandyopadhyay, editors, *Proc. of the ECS 98-19 Quantum Confinement V: Nanostructures*, pages 574–581. Electrochemical Society, 1999.

[40] F. H. Bennett III, J. R. Koza, J. Shipman, and O. Stiffelman. Building a parallel computer system for $18,000 that performs a half-petaflop per day. In W. Banzhaf, J. Daida, A. E. Eiben, et al., editors, *Proc. Genetic and Evolutionary Computation conference (GECCO-99)*, pages 1484–1490. Morgan Kaufmann, 1999.

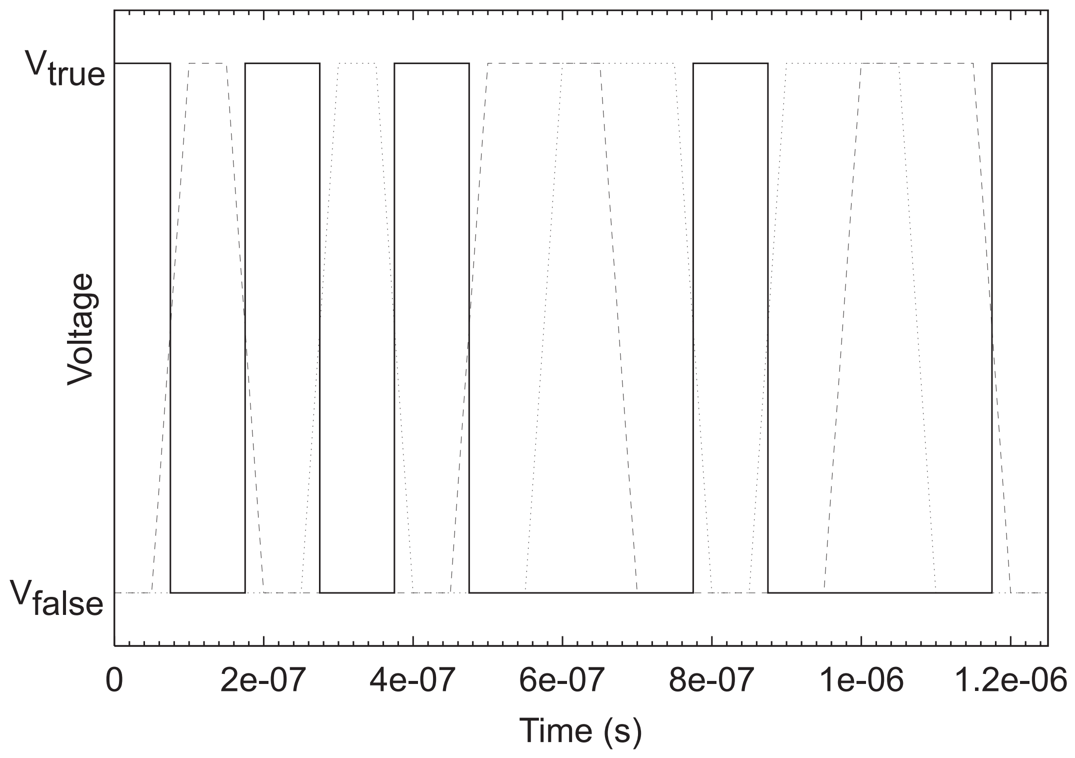

Figure 1: The input waveforms for a fitness evaluation (dotted and dashed) and the ideal output $V_{ideal}(t)$ (solid).

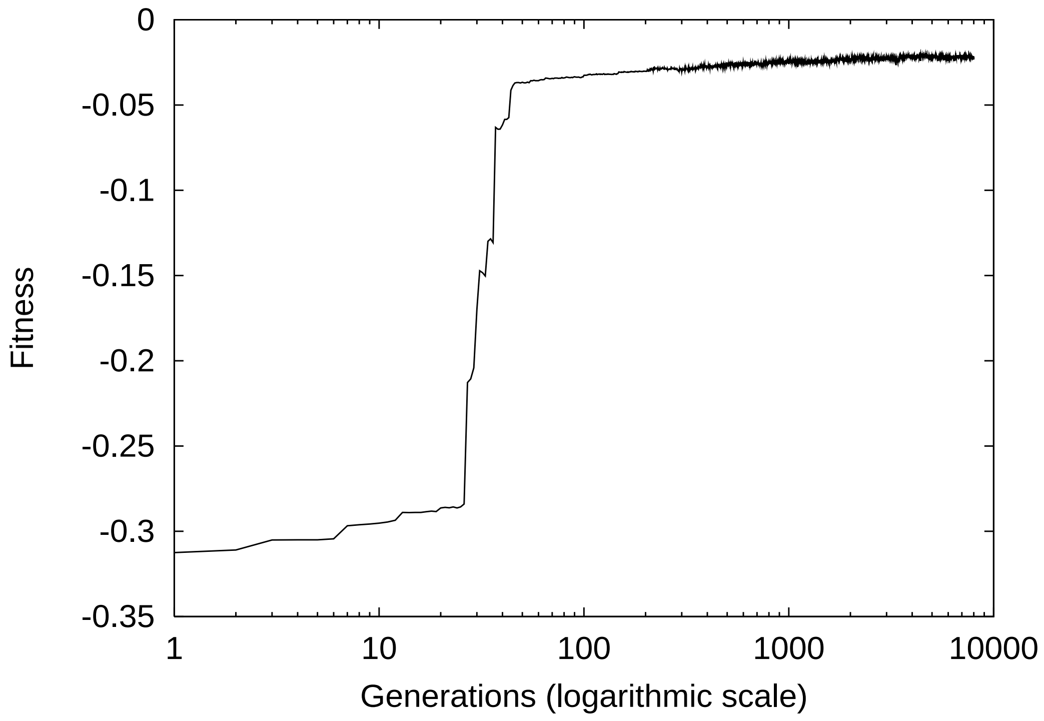

Figure 2: Evolution at 0K: Fitness of the best individual in the population.

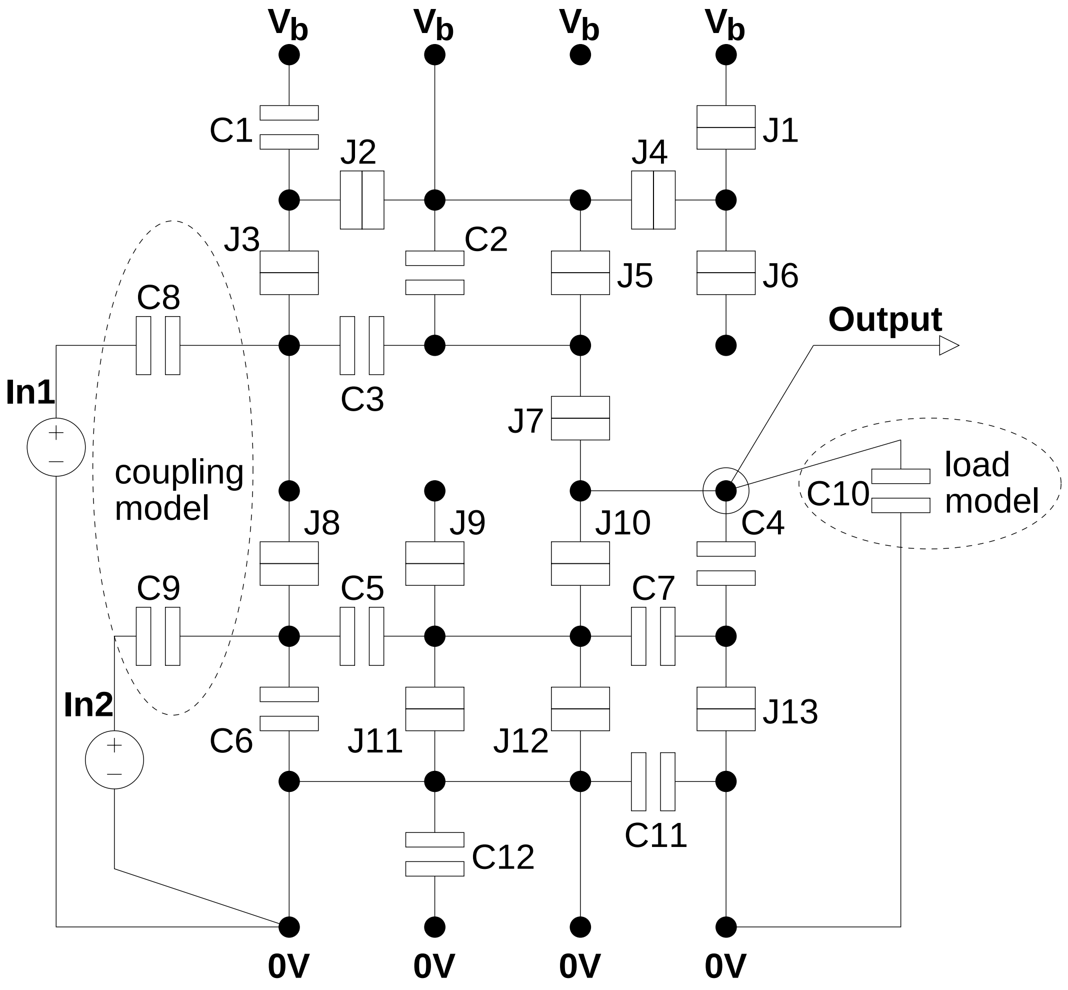

Figure 3: The circuit evolved at 0K, in the representation manipulated by the evolutionary algorithm. The large dots are nodes, and the 'preferred output' node is shown circled.

Fixed Values:		
C8	3.333e-13 F	
C9	3.333e-13 F	
C10	1.000e-12 F	
Evolved Values:		
C1	6.416e-19 F	
C2	4.001e-19 F	
C3	5.387e-19 F	
C4	4.645e-17 F	
C5	9.597e-16 F	
C6	4.638e-18 F	
C7	1.000e-13 F	
J2	9.462e-16 F	2.756e+05 Ω
J3	2.466e-16 F	3.114e+05 Ω
J5	8.867e-14 F	2.900e+08 Ω
J7	9.861e-18 F	9.571e+08 Ω
J8	1.472e-15 F	2.668e+08 Ω
J10	3.946e-18 F	5.000e+04 Ω
J11	4.000e-19 F	5.000e+04 Ω
J12	4.000e-19 F	5.024e+04 Ω
J13	4.000e-19 F	8.902e+07 Ω
V_b	-1.000e-04 V	
V_{false}	-1.538e-05 V	
V_{true}	-9.923e-05 V	

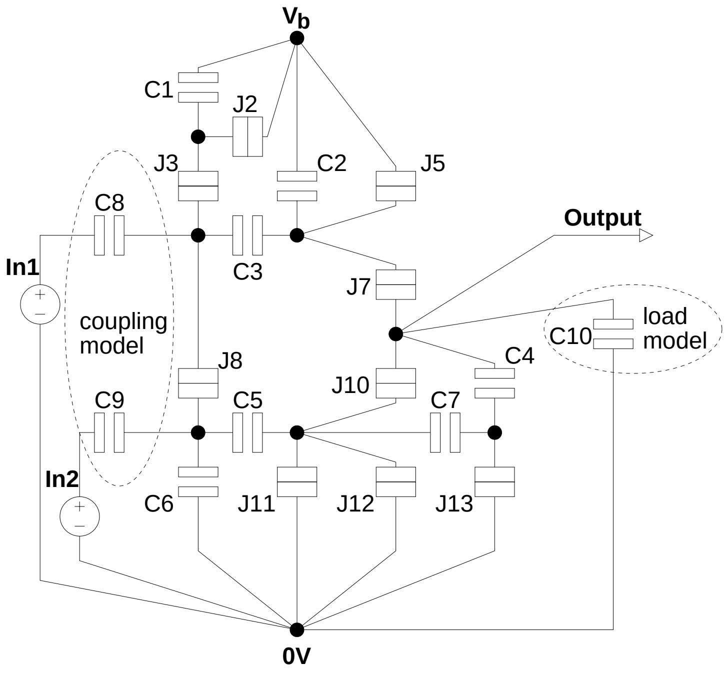

Figure 4: The circuit evolved at 0K, as seen by the simulator: Nodes joined by 'virtual wires' have been amalgamated, and shorted or dangling components have been removed.

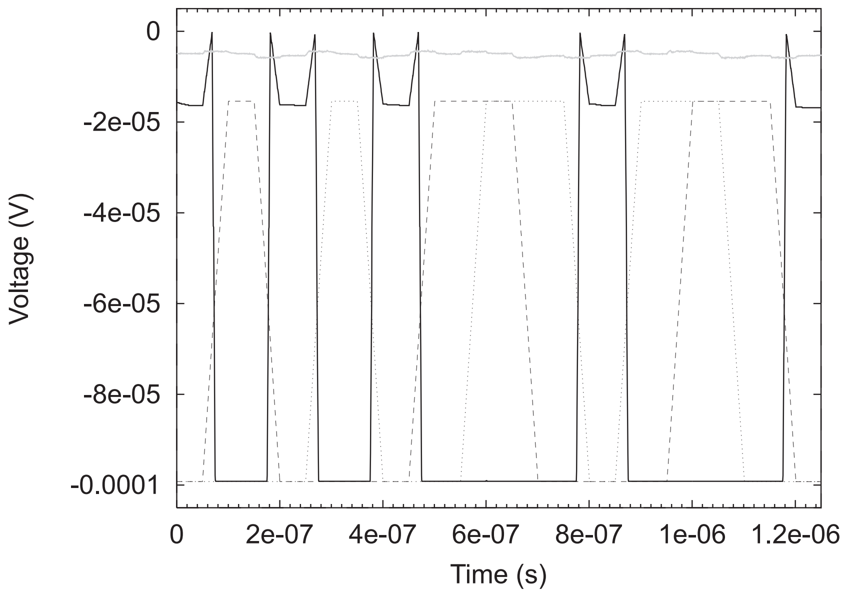

Figure 5: The input/output relationship of the circuit evolved at 0K (see Fig. 4). The dark solid line is the output considering only first order tunnelling events, as used in the simulations to evaluate fitness during evolution. The gray solid line is the output when second-order (or second and third-order) events are also simulated.

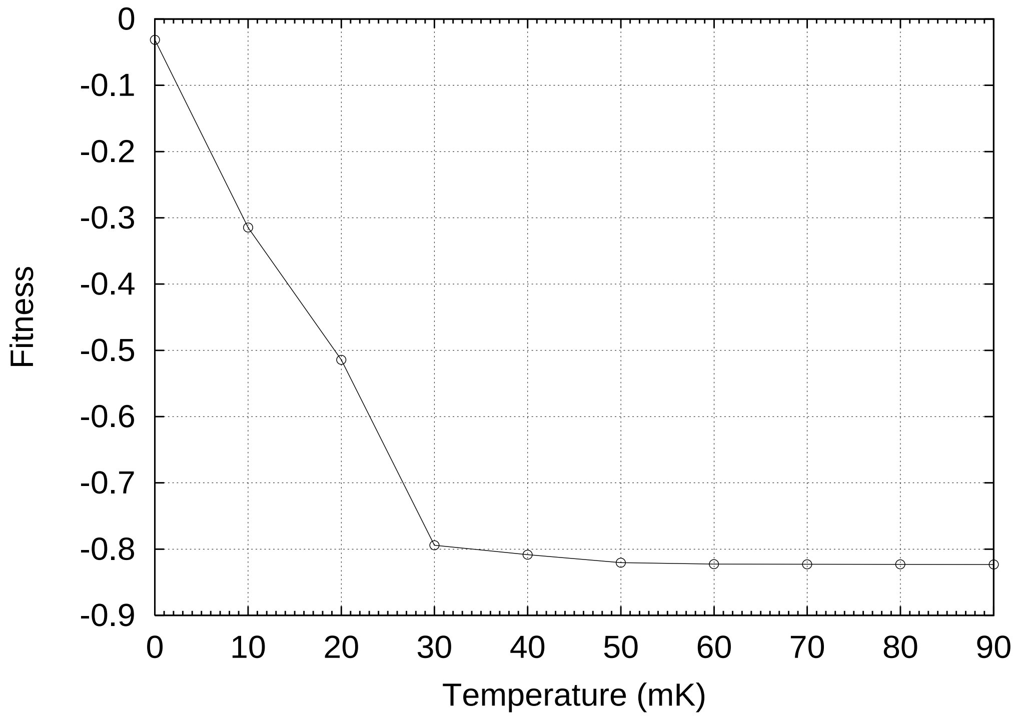

Figure 6: The thermal response of the circuit evolved at 0K (see Fig. 4).

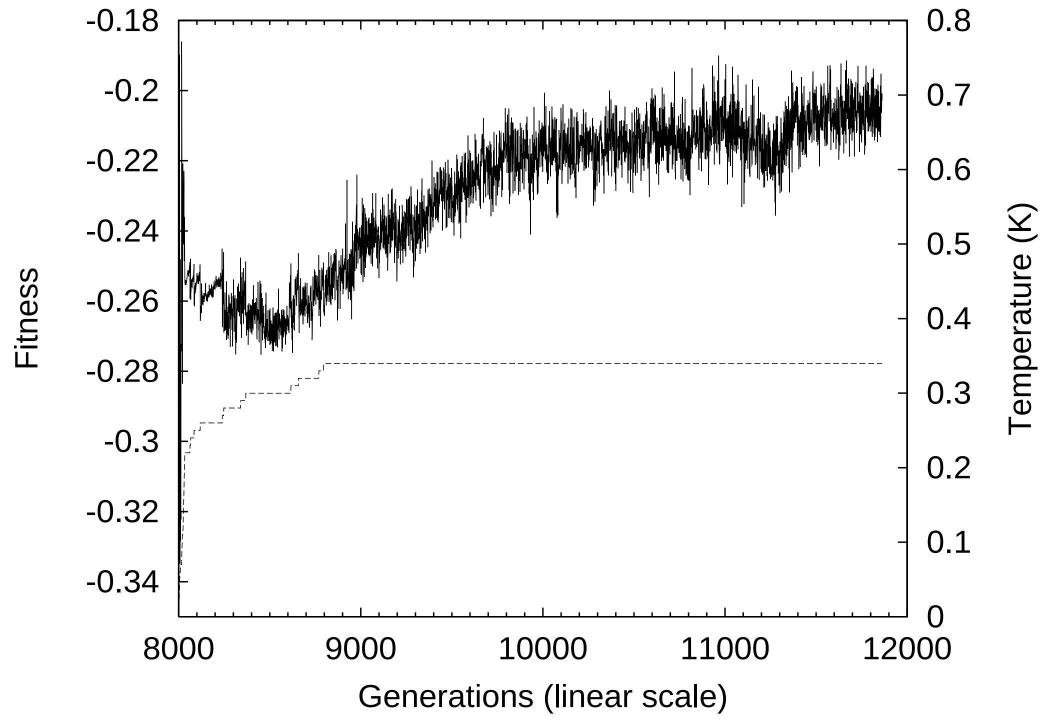

Figure 7: Continued evolution, at increasing temperature. The solid upper line is best fitness (left axis), and the lower dotted line is temperature (right axis). The temperature was increased by 10mK whenever the fitness reached -0.25, then was held constant upon reaching 340mK.

Fixed Values:		
C2	1.000e-12 F	
C8	3.333e-13 F	
C9	3.333e-13 F	
Evolved Values:		
C1	4.858e-19 F	
C3	9.969e-14 F	
C4	2.052e-16 F	
C5	1.000e-13 F	
C6	3.393e-16 F	
C7	2.975e-15 F	
J1	4.000e-19 F	4.950e+06 Ω
J2	4.000e-19 F	5.766e+05 Ω
J3	4.059e-19 F	9.024e+04 Ω
J4	4.237e-19 F	5.854e+04 Ω
J5	3.632e-16 F	2.886e+07 Ω
J6	4.857e-19 F	5.000e+04 Ω
V_b	-1.000e-04 V	
V_{false}	-8.368e-05 V	
V_{true}	-8.488e-06 V	

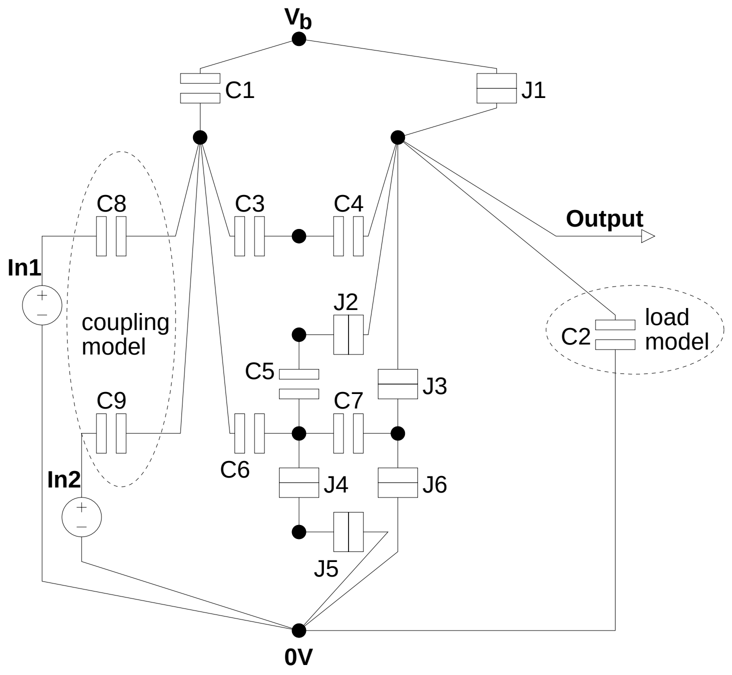

Figure 8: The best circuit so far for 340mK.

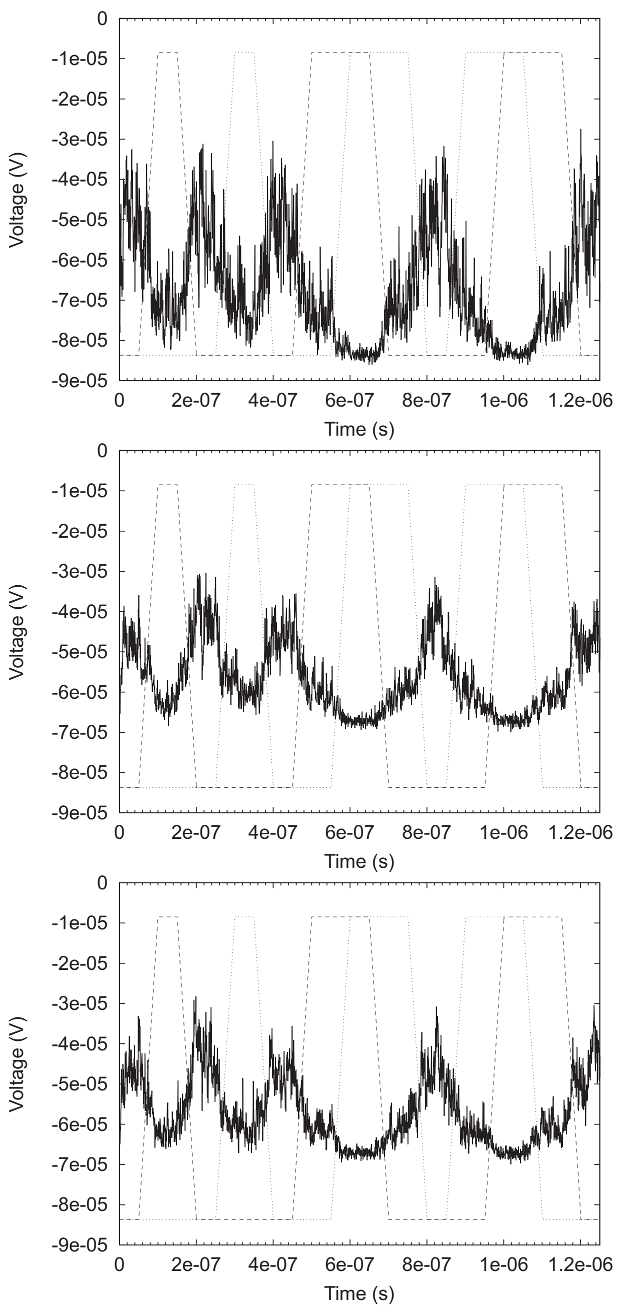

Figure 9: The input/output relationship of the circuit evolved at 340mK (see Fig. 8). **Top:** Simulation only of first-order tunnelling events. **Middle:** Simulation including second-order tunnelling. **Bottom:** Simulation including third-order tunnelling.

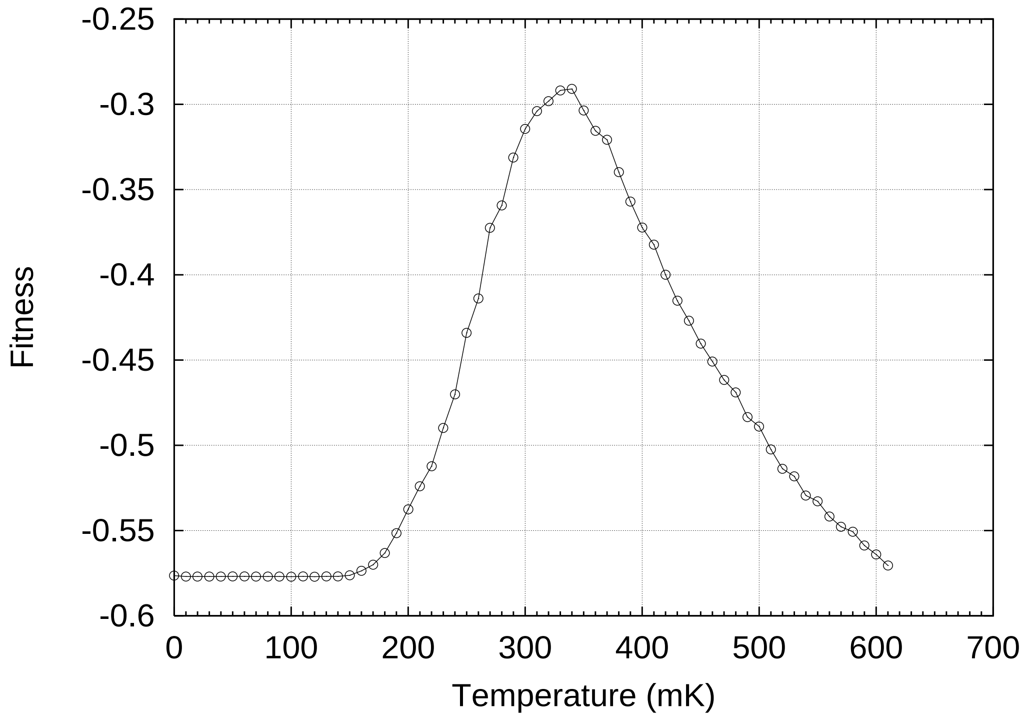

Figure 10: The thermal response of the circuit evolved for 340mK (see Fig. 8).

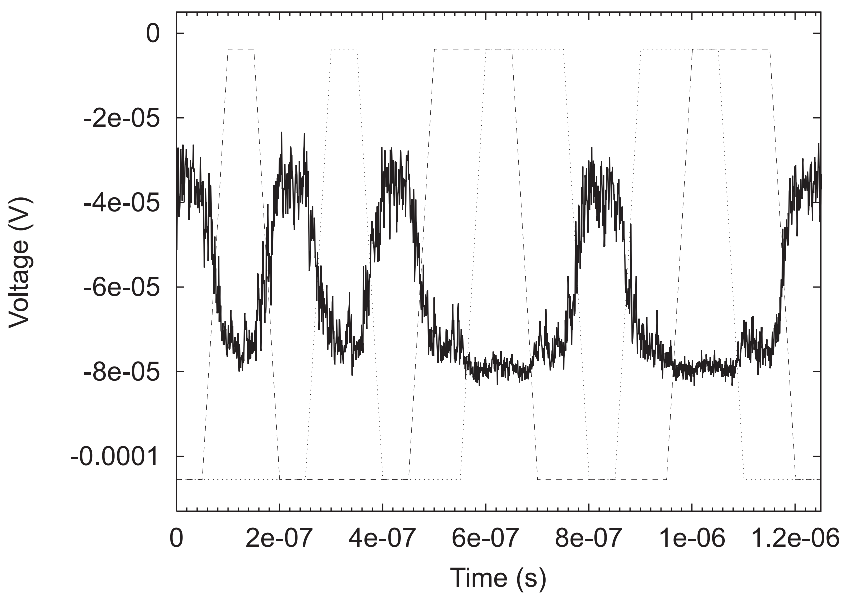

Figure 11: An improved input/output relationship at 340mK, but only if multiple tunnelling events are neglected.