Manipulating the conductance of single-walled carbon nanotubes based thin films for evolving threshold logic circuits using particle swarm optimisation

F. Qaiser, A. Kotsialos, *Member, IEEE*, M. K. Massey, D. A. Zeze, C. Pearson and M. C. Petty
School of Engineering and Computing Sciences, Durham University, Durham, United Kingdom
E-mail:{fawada.qaiser, apostolos.kotsialos, m.k.massey, d.a.zeze, christopher.pearson, m.c.petty}@durham.ac.uk

***Abstract*—Evolution In Materio (EIM) is concerned with solving computational problems by exploiting the physical properties of materials. This paper presents the results of using a particle swarm optimisation (PSO) algorithm for evolving logic circuits in single-walled carbon nanotubes (SWCNT) based composites on a special custom made platform. The material used is a composite of SWCNT dispersed randomly in a polymer forming a complex conductive network. Following the EIM methodology the conductance of the material is manipulated for evolving threshold based logic circuits. The problem is formulated as a constrained, mixed integer optimisation problem. It is solved using PSO in conjunction with the shortest position value rule. The results showed that the conductive properties of SWCNT can be used to configure these materials to evolve multiple input/ output logic circuits.**

1. Introduction

Unconventional computing aims at investigating methods for designing and developing systems that are able to perform a computation in different ways than the current paradigm. One such direction of research is evolution in materio (EIM), which is concerned with computing performed based on different materials. EIM focuses on the underlying properties of the materials aiming at exploring and exploiting them in such a way so that they are brought to a computation inducing state. Contrary to traditional computing with MOSFET technology, where everything is top-down designed, produced and programmed very carefully, EIM uses a bottom up approach where computation is performed by the material without having explicit knowledge of its internal properties.

Figure 1 illustrates EIM's concept. An optimisation algorithm selects a set of incident signals applied on to the material, changing in effect its physical properties. At each iteration, the material is tested against a number of known input/output pairs of a successful computation. The material's response is recorded for each of those test inputs and a global error function is evaluated. An optimisation algorithm manipulates the material's properties within an iterative search by applying different configuration signals. This evolutionary process brings the material to a state where it can perform the desired computation in the sense that its outputs can be interpreted according to a prespecified scheme.

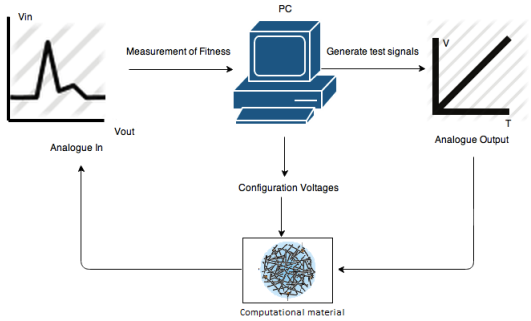


Figure 1: Concept of evolution in materio [1].

The idea of EIM is based on the observations made in [2], where evolutionary algorithms were used for designing electrical circuits on FPGAs. The resulting circuit topologies were influenced by the material of the board used. The reason for this is that optimisation methods when applied on material systems are in effect feedback systems. Their pattern of search is affected by the collected information, which includes the material properties of the particular equipment used. Hence, solutions based on a specific FPGA would depend on properties that were unaccounted for during the board's design. EIM replaced the FPGAs with material systems, which are viewed as black boxes. This approach favours using all inherent physical properties of the material medium [3] for a search algorithm to achieve its goals.

This evolutionary approach requires a platform that provides access to physical properties of the material in use and can bridge the gap between their analogue nature and the digital nature of a computer supervising an evolutionary search. In [4], an *mbed* platform was used for evolving threshold logic circuits using population based optimisation. Based on this work a more detailed study on the material properties and its capability of performing a computation was conducted and results are reported in [5], [6] and [7]. The hardware used for these studies was relatively inflexible and did not allow the use of algorithms with extended vectors of decision variables regarding the selection of an incident signal's location of application on the material body. Here, a more powerful and versatile platform is used

which allows for a more flexible problem formulation to be realised. This is the mecobo board [8], which is a purpose built platform that can interface with large variety of materials and also has the flexibility to control and map variety of input, output and configuration signals and their properties. In addition to the different hardware used, this paper extends the work reported in [4], [5] and [6] by using a particle swarm optimisation (PSO) instead of the Nelder-Mead and Differential Evolution algorithms used there.

2. Materials

EIM is based on materials that are configurable by the applied signals [9]. Different materials that have been followed include biological material like slime moulds [10], [11], bacterial consortia [12] and biological cells (neurons) [13] as well as non-biological materials such as, liquid crystals [14], single-walled carbon nanotubes (SWCNT) [4], nano-particles [15]. SWCNT based materials have shown the potential to solve variety of computational problems [4] [16] [17] [18] [19].

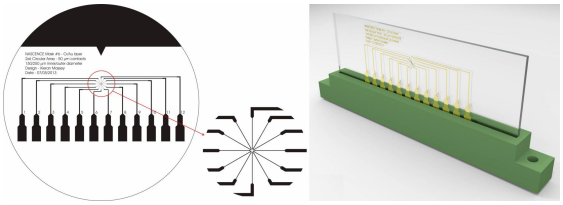
The materials used in this study are composites of SWCNTs and poly(methylmethacrylate) (PMMA). The concentration of SWCNTs (as a weight percentage of the PMMA) and the viscosity of the PMMA / SWCNT mixture affect the material behaviour. The samples are prepared by dispersing SWCNTs in Anisole (VWR, analytical reagent grade) with an aid of an ultrasonic probe at a power of 20% (Cole-Parmer $750W$ ultrasonic homogeniser). The PMMA (Aldrich, $M_w = 93,000$) is then added and additional sonication was performed for a more uniform dispersion. The material was then deposited on gold micro-electrode arrays by drop casting. In order to promote quick drying and a more uniform coverage, the substrate was heated to 100°C and left for 30 minutes in order to dry any remaining solvent.

Micro-electrode arrays are fabricated from gold on standard borosilicate glass substrates, using etch-back photolithographic techniques. The arrays are designed with very small electrode separations ($22\mu m$) so that high strength electric fields ($5 \times 10^5 V/m$) can be applied even with the modest voltages ($10.8V$). Figure 2b shows an optical micrograph of the SWCNT material deposited on the electrodes.

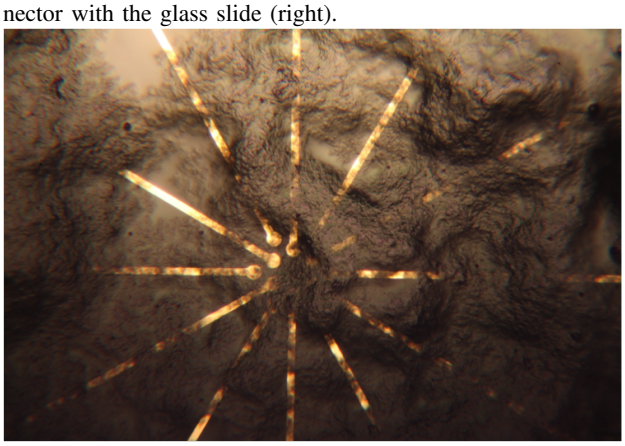
3. The mecobo hardware platform

The mecobo platform can be connected to a host computer over a USB as a standalone interface. The host computer runs the system interface as well as the optimisation algorithm. The software interface configures the hardware by providing the necessary information for each connection with the material. Its response to sequences of test inputs is recorded and transformed into appropriate data format for the calculation of the error function.

Mecobo is designed as a PCB board with an FPGA and a microcontroller as main components, Figure 3. Digital-to-Analogue and Analogue-to-Digital Converters, DAC and



(a) Electrode photolithography mask (left) and PCB edge connector with the glass slide (right).



(b) Optical micrograph of carbon nanotubes deposited on gold electrode arrays.

Figure 2: Microelectrode arrays and SWCNTs deposited on gold micro-electrode arrays.

ADC, respectively, are used for generating signals and collecting measurements from the electrodes resting on the material. The microcontroller accepts the commands from the host computer and implements them in FPGA via shared memory. The FPGA establishes the physical and logical communication with the material. Digital I/O produce signals and sample responses. Analogue output signals are produced by the DAC modules, which can produce static or time dependent voltage waveforms. The ADC modules perform the sampling of analogue waveforms received from the material.

Furthermore, a scheduler, Figure 4, is implemented in the hardware that can schedule the time slots for different I/O or configuration signals or to compensate delays when materials need time to settle before any meaningful computation can be observed. Figure 4 illustrates the interface hardware implementation.

The SWCNT are randomly distributed forming an inhomogeneous random network of nanotube bundles. This material is spread over the microelectrode arrays and the input/output locations can be aribitrary. The choice of input/output and configuration electrode terminals are left to be decided by the optimisation algorithm. In order to achieve this, a pin routing module is placed between signal generating modules and the sampling buffer. Hence, in contrast to previous experiments, where pin configuration was predetermined, the experiments presented in this paper implement variable pin configuration that is under the control of the

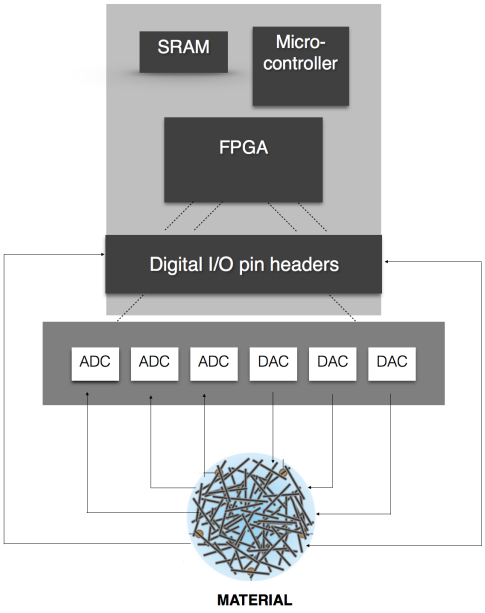


Figure 3: Mecobo block diagram for EIM.

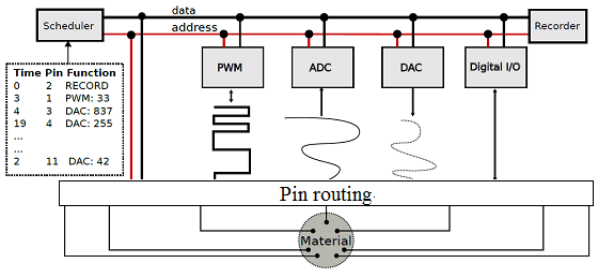


Figure 4: Mecobo scheduler [8].

optimisation algorithm.

Different computational problems have been suggested for such a system. The calculation of Boolean functions based on threshold logic is considered here. A more detailed discussion on candidate problems can be found in [20].

4. Threshold logic gates

Threshold logic gates operate on the principle of comparing the output of a device to a prespecified threshold for deciding if it is at a logical 0 or 1 state. Such a threshold logic device has n inputs $A_i \in \{0,1\},\ i = 1,\dots,n$ and m outputs $Y_j \in \{0,1\},\ j = 1,\dots,m$. For output y_j, a number $L_j + 1$ of thresholds $\theta_{j,p},\ p = 0,\dots,L_j$ are used for differentiating between possible inputs. y_j is decided by a rule of the following form:

$$y_j = Y_{j,p} \text{ if } \theta_{j,p-1} \leq \sum_{i=1}^{n} w_i A_i < \theta_{j,p}, \qquad (1)$$
$$j = 1,\cdots,m,\ p = 1,\dots,L_j.$$

The detailed design of a device like this requires the complete determination of the weights $w_{j,p}$ and the corresponding threshold levels $\theta_{j,p}$. This is achieved by using circuit design tools. Instead of following this detailed approach, a different operational principle may be adopted, whereby the device is a piece of inhomegeneous conducting material where a number of inputs are applied directly to its body and a number of output measurements are collected from it.

There are two types of input applied in this device, q configuration inputs $V_z,\ z = 1,\dots,q$ and n computation arguments $x_i,\ i = 1,\dots,n$. Within the EIM framework, configuration inputs are used for changing the material's properties of interest and bring it to a computation inducing state. A material state is characterised as such when the material's response can be uniquely translated to the correct computation when an arbitrary set of input arguments are applied to its body.

Let q denote the number of configuration inputs $V_z,\ z = 1,\dots,q$, organised into vector $\mathbf{V}$ and $\mathbf{x} \in X \subset \mathbb{R}^n$ the space of possible inputs used for representing a binary vector $\mathbf{A} \in \{0,1\}^n$, i.e. there is a one-to-one mapping $\mathbf{R} : X \to \{0,1\}^n$. $\mathbf{R}(\mathbf{x}) = \mathbf{A}$ means that $\mathbf{x} \in X$ uniquely represents binary input $\mathbf{A} \in \{0,1\}^n$. The material's measured response when q configuration and n computation argument inputs are applied, at output location j is $M_j(\mathbf{x},\mathbf{V})$. A threshold logic gate based on M_j can be obtained by modifying eqn. (1) as follows.

$$y_j = Y_{j,p} \text{ if } \theta_{j,p-1} \leq M_j(\mathbf{x},\mathbf{V}) < \theta_{j,p}, \qquad (2)$$
$$j = 1,\dots,m,\ p = 1,\dots,L_j.$$

The detailed design effort required for obtaining the weights $w_{j,p}$ is replaced by a search process that aims at identifying a suitable $\mathbf{R}$ and finding the values of x_i, V_z and $\theta_{j,p}$. Making the material respond in a unique way to a given combination of configuration and (arbitrary in the domain of definition) computation argument inputs, is the task of EIM.

For the particular type of material used here, the output M_j is an increasing function of the total input, $\mathbf{x}$ and $\mathbf{V}$. The principle of operation is based on the implementation of rule (2) by dividing the output from location j into bands defined by thresholds $\theta_{j,p}$. The computation is then performed by mapping a particular input to the corresponding output band, which indicates a logical 0 or 1 consistent with the Boolean function truth table. The truth table $\mathbf{Y}(\mathbf{A}) \in \{0,1\}^m$, $A \in \{0,1\}^n$, dictates the value of Y_j when a vector $\mathbf{x}$ representing $\mathbf{A}$ is applied to the material while it is in the computation inducing state. This is shown in Figure 5 where rule (2) is graphically represented.

5. Material training

The material at hand is a randomly dispersed network of SWCNTs acting as resistors. The configuration and argument inputs are constant voltage pulses applied at different locations on the material's body and the outputs M_j are voltage measurements collected from other locations. The

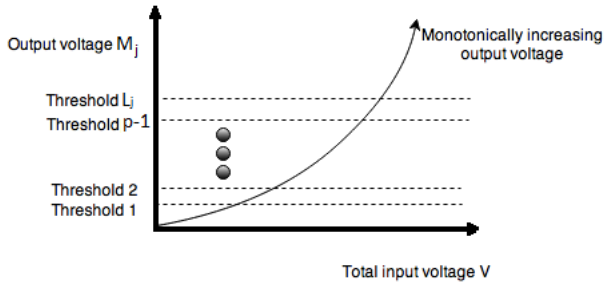


Figure 5: Threshold based logic for a material output.

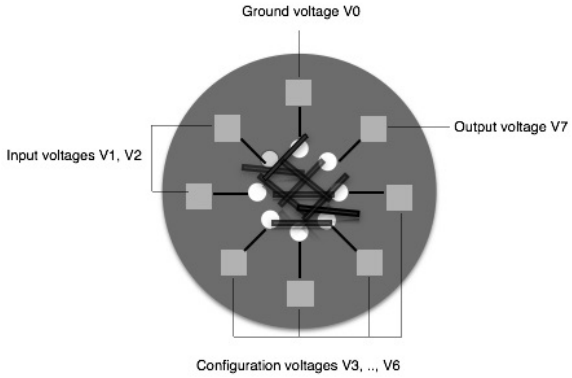


Figure 6: An example of arrangement of input, output and configuration electrodes.

argument inputs are short voltage pulses applied *while* a particular set of configuration inputs are applied. The sampling frequency of the output measurements is the same as the argument's inputs. Mecobo allows the parametrisation of the pins used for applying voltage inputs and those used for output measurements. Hence, the problem formulation is an extension of the one proposed in [4] in the sense that the optimisation algorithm is allowed to select the pin assignment. Figure 6 depicts an instant of a particular assignment for a two-input single-output gate. There are eight locations where electrodes are connected with the material; each one can be selected to be an input or an output (location 0 is always the ground). In the instant shown in Figure 6, locations 1 and 2, where voltages V_1 and V_2 are applied, correspond to the gate's input terminals. The configuration voltages are applied at electrode terminals 3–6; these are used for manipulating the material's conductivity so that the measured output can be interpreted according to rule (2). The output is measured at location 7 where voltage V_7 is collected, i.e. $M_1(\mathbf{x}, \mathbf{V}) = V_7$.

Let $\mathbf{P} = \left[P_1 \dots P_{n+q+m}\right]^T$ denote the vector describing the pin assignment. $P_\ell \in \{1, \dots, q+n+m\}$ is the pin number at position ℓ. The first n positions in $\mathbf{P}$ correspond to the argument inputs, the next q to the configuration voltages and the last m to the outputs. For the example of Figure 6, $\mathbf{P} = \left[1, 2, 3, 4, 5, 6, 7\right]^T$, with the pins' numbering starting where V_1 is applied and increasing in the counter-clock wise direction.

The material training problem at hand aims at identifying the optimal pin assignment $\mathbf{P}^*$, configuration voltages $\mathbf{V}^*$ and vector of thresholds θ^* for a given mapping $\mathbf{R}$ such that when the $\mathbf{V}^*$ are concurrently applied, the application of any input voltage $\mathbf{x}$ results to measurements M_j; when rule (2) using θ^* is applied, the equation $\mathbf{y}\left(\mathbf{P}^*, \mathbf{R}(\mathbf{x}), \mathbf{V}^*, \theta^*\right) = \mathbf{Y}\left(\mathbf{R}(\mathbf{x})\right)$ is true ($\mathbf{y}$ is the vector of y_j).

A set of K training data is $\left(\mathbf{A}^{(k)}, \mathbf{Y}(\mathbf{A}^{(k)})\right)$, $k = 1, \dots, K$ covering all possible combinations of binary inputs/outputs is generated. The objective function J of the optimisation problem is

$$J = \sum_{k=1}^{K} \left[\mathbf{y}\left(\mathbf{R}\left(\mathbf{x}^{(k)}\right), \mathbf{V}, \theta\right) - \mathbf{Y}\left(\mathbf{A}^{(k)}\right)\right]^2 \quad (3)$$

where

$$\mathbf{x}^{(k)} = R^{-1}\left(\mathbf{A}^{(k)}\right). \quad (4)$$

J is calculated by interacting with the material based on a given pin assignment. In the most general case, the vector of decision variables $\mathbf{B}$ has the form

$$\mathbf{B} = \left[\mathbf{P}^T\ \mathbf{x}^T\ \mathbf{V}^T\ \theta^T\right]^T. \quad (5)$$

A simple mapping $\mathbf{R}$ is obtained by setting $X = \{0, 1\}$ and $A_i = 0$ is represented by $x_i = 0$ V and $A_i = 1$ by $x_i = 1$ V. In this case $\mathbf{x}$ is fixed and therefore not part of $\mathbf{B}$ in (5). The case where $\mathbf{x}$ a function of $\mathbf{V}$ is discussed in [4].

The training optimisation problem is that of minimising (3) subject to (4), (2) and the following simple bound constraints on the configuration voltages $\mathbf{V}$.

$$V_{z,\min} \le V_z \le V_{z,\max},\ z = 1, \dots, q. \quad (6)$$

Notice that there are no explicit constraints for the thresholds θ. The optimisation algorithm discards those solutions where the ordering necessary for (2) to work is not held.

This optimisation problem is solved using a classical PSO algorithm, [21]. However, the $\mathbf{P}$ part of the vector of decision variables poses a permutation problem as a pin can only be used for a single input or output. This is dealt with by using the shortest position value (SPV) rule as proposed in [22] and [23]. The SPV rule converts the continuous position generated by PSO to a discrete value. The SPV rule is applied in two phases, one when the particles are generated and the other when the particles are updated. The rest of decision variables are considered as continuous.

Although $\mathbf{V}$ is continuous for the PSO algorithm, the implementation of a particular value goes through the Mecobo board, which can apply discrete levels of voltage. For all pins, the minimum voltage is $V_{\min} = -5.0$ V and the maximum is $V_{\max} = +5.0$ V. This voltage range is divided into 255 equidistant discrete level values with 0 corresponding to -5.0 V and 255 to $+5.0$ V.

TABLE 1: Truth table for $(A_1 + A_2 + A_3) \oplus (A_1A_2A_3)$.

Inputs			Output
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

6. Results and discussion

A number of Boolean functions have been used for making the material yield a threshold logic circuit in the sense described in section 4. The material used had a concentration of 0.1% SWCNT, see section 2. Simple AND and OR gates were easily obtained. Here, the results are discussed for a 3-input 1-output logic circuit, the half-adder and the full-adder.

An initial trial of applying random configuration voltages resulted to outputs that could not be translated into computational outcomes when each circuit's threshold rule was applied. The same was true when a circuit of resistors was used instead of the SWCNT composite. It is the PSO algorithm that identifies suitable values for those configuration inputs and thresholds. In other words, the material in its initial state cannot perform the required calculation. It should be noted, that no shape change takes place during training. The only physical property affected is the material's conductance. The material itself is very stable and the results reproducible.

6.1. Logic circuit $(A_1 + A_2 + A_3) \oplus (A_1A_2A_3)$

This is a three input, one output logic circuit based on AND, OR and XOR. Its truth table is shown in Table 1. At least one and at most two inputs need to be true in order for the output to be true. If all inputs are the same, either true or false, the output is false.

Referring to the problem formulation of section 5, $n = 3$, $m = 1$ and $q = 8$. Two thresholds, θ_1 and θ_2 are required to separate between the output true and false states, one for distinguishing an all false inputs state and another for the all true inputs. The rule implemented is

$$Output = \begin{cases} 1 & \text{if } \theta_1 < M_1(\mathbf{x}, \mathbf{V}) < \theta_2 \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

The optimal solution the PSO algorithm converged to is given in Table 2. It provides the optimal configuration voltages as well as the threshold values and pin assignment. Figure 7 shows the output measurements and optimal thresholds for all possible binary inputs. These were obtained from measuring the voltage at pin 10, while the material was constantly charged with the optimal configuration voltages at pins 2, 11, 8, 0, 7, 9, 1, 4, and a random sequence of binary triplets was applied to pins 5, 6 and 3. The output measurements between the two thresholds are interpreted as logic 1 and outside of it as 0.

TABLE 2: Optimal solution for logic circuit $(A_1 + A_2 + A_3) \oplus (A_1A_2A_3)$

Thresholds	$\theta_1 = -3.64$, $\theta_2 = -0.86$
Configuration voltages	$V_1 = -1.64$,$V_2 = -2.68$,$V_3 = -2.87$ $V_4 = -3.54$,$V_5 = -2.34$, $V_6 = -1.81$ $V_7 = -1.89$, $V_8 = -2.19$
Pin assignment (signal) $\rightarrow$) (pin#)	$x_1 \rightarrow 5$, $x_2 \rightarrow 6$, $x_3 \rightarrow 3$ $M_1 \rightarrow 10$, $V_1 \rightarrow 2$, $V_2 \rightarrow 11$ $V_3 \rightarrow 8$, $V_4 \rightarrow 0$,$V_5 \rightarrow 7$ $V_6 \rightarrow 9$, $V_7 \rightarrow 1$, $V_8 \rightarrow 4$

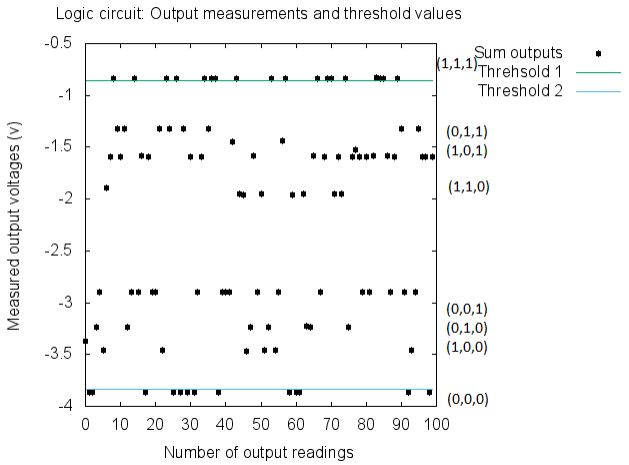


Figure 7: Material response for the $(A_1 + A_2 + A_3) \oplus (A_1A_2A_3)$ circuit; output measurements and thresholds.

6.2. Half-adder

The half-adder is a two-input two-output circuit using an XOR and an AND output. Its truth table is shown in Table 3.

For the training optimisation problem, $n = 2$, $m = 2$ and $q = 8$. The two output measurements M_1 and M_2 are collected for the XOR and AND, respectively, for any binary pair (A_1, A_2). In order to make the material behave as a half-adder based on the use of thresholds, the output corresponding to AND, M_1, requires the use of a single threshold and the XOR measurement, M_2, needs two. For M_1 and the AND carry output,

$$Output\ Carry = \begin{cases} 1 & \text{if } M_1(\mathbf{x}, \mathbf{V}) \geq \theta_1 \\ 0 & \text{otherwise.} \end{cases} \quad (8)$$

Hence, a measured M_1 larger than θ_1 is interpreted as a logical 1, otherwise as a 0.

TABLE 3: Truth table for half adder

Inputs		Output sum (XOR)	Output carry (AND)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

TABLE 4: Optimal solution for half-adder circuit.

Threshold for carry	$\theta_1 = -1.4$
Thresholds for sum	$\theta_2 = -2.9$, $\theta_3 = -1.9$
Configuration voltages	$V_1 = 0.03$, $V_2 = 1.45$, $V_3 = 3.19$ $V_4 = -0.21$, $V_5 = -0.75$, $V_6 = -0.68$ $V_7 = -0.53$, $V_8 = -1.75$
Pin assignment (signal) $\rightarrow$) (pin#)	$x_1 \rightarrow 7$, $x_2 \rightarrow 6$, $M_1 \rightarrow 1$ $M_2 \rightarrow 3$, $V_1 \rightarrow 9$, $V_2 \rightarrow 4$ $V_3 \rightarrow 2$, $V_4 \rightarrow 10$, $V_5 \rightarrow 5$ $V_6 \rightarrow 0$, $V_7 \rightarrow 11$, $V_8 \rightarrow 8$

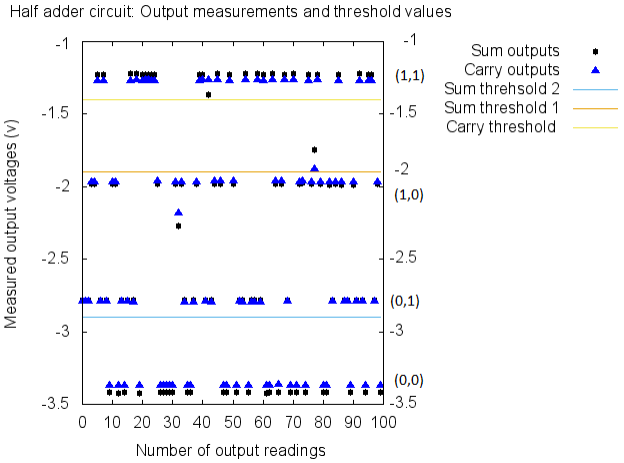


Figure 8: Material response for the half-adder circuit; output measurements and thresholds.

For M_2 and the XOR sum output,

$$Output\ Sum = \begin{cases} 1 & \text{if } \theta_2 \leq M_2(\mathbf{x}, \mathbf{V}) \leq \theta_3 \\ 0 & \text{otherwise.} \end{cases} \quad (9)$$

Hence, a measured M_2 between the two thresholds θ_2 and θ_3 is interpreted as a logical 1 at the sum output, otherwise if it is below θ_2 or above θ_3 as a logical 0.

Training the material based on eqns. (8) and (9) for interpreting the measured outputs as half-adder circuit computation results to the solution shown in Table 4.

The output measurements at two electrodes are shown in Figure 8. The three thresholds can be differentiated clearly. One is used for calculating the carry and two for the sum. The electrode pins 7, and 6 are chosen as input terminals and 1 and 3 as output; the rest of the pins are configuration voltage terminals. It should be noted that the measurements M_1 and M_2 are taken concurrently, while the configuration voltages are constantly been applied.

6.3. Full-adder

The full-adder circuit has three inputs and two outputs, the sum and carry, hence $n = 3$ and $m = 2$ leaving $q = 7$ configuration voltages available for obtaining a solution. Again, two measurement pins are used, M_1 for the carry and M_2 for the sum. The circuit's truth table is given in Table 5.

TABLE 5: Truth table for the full-adder circuit.

Inputs			Output sum	Output carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

TABLE 6: Optimal solution for the full-adder circuit

Threshold for carry	$\theta_1 = -2.9$
Threshold for sum	$\theta_2 = -3.9$, $\theta_3 = -3.4$, $\theta_4 = -0.9$
Configuration voltages	$V_1 = 0.03$, $V_2 = -0.62$, $V_3 = -1.99$ $V_4 = -2.17$, $V_5 = -2.82$, $V_6 = -1.97$ $V_7 = -1.74$
Pin assignment (signal) $\rightarrow$) (pin#)	$x_1 \rightarrow 4$, $x_2 \rightarrow 1$, $x_3 \rightarrow 2$ $M_1 \rightarrow 7$, $M_2 \rightarrow 5$, $V_1 \rightarrow 10$ $V_2 \rightarrow 6$, $V_3 \rightarrow 3$, $V_4 \rightarrow 9$ $V_5 \rightarrow 0$, $V_6 \rightarrow 11$, $V_7 \rightarrow 8$

A single threshold is used for the carry leading to the following interpretation of measurement M_1.

$$Output\ carry = \begin{cases} 1 & \text{if } M_1(\mathbf{x}, \mathbf{V}) \geq \theta_1 \\ 0 & \text{otherwise.} \end{cases} \quad (10)$$

The sum operation is more complicated and requires three thresholds to calculate it at the output terminal where M_2 is measured. This leads to the following threshold rule.

$$Output\ sum = \begin{cases} 0 & \text{if } M_2(\mathbf{x}, \mathbf{V}) < \theta_2 \\ 1 & \text{if } \theta_2 \leq M_2(\mathbf{x}, \mathbf{V}) < \theta_3 \\ 0 & \text{if } \theta_3 \leq M_2(\mathbf{x}, \mathbf{V}) < \theta_4 \\ 1 & \text{if } M_2(\mathbf{x}, \mathbf{V}) \geq \theta_4 \end{cases} \quad (11)$$

The optimal solution is given in Table 6. Seven configuration voltages were used with three inputs and two outputs. The three inputs are applied at pins 4, 1 and 2, whereas the outputs are collected from pins 7 and 5. The material's response to each of the specific 16 possible binary inputs is shown in Figure 9, along with the thresholds used. Compared to the half-adder response shown in Figure 8, the full-adder measured output is more complicated because of the three, rather than two, binary inputs. The optimisation is able to find a solution, where the material's response is organised so that the interpretation scheme followed results to the full-adder outcome.

7. Conclusion

This paper presents the results of applying PSO for EIM where the material consists of a SWCNTs/PMMA mixture. By manipulating the material's conductance, it is possible to perform calculations of Boolean functions based on a threshold logic interpretation scheme. The mecobo platform allowed the flexibility of selecting the input, output and configuration terminals and put them under the control of an optimisation algorithm. The experiments demonstrated mecobo's functionality and suitability as an interface for

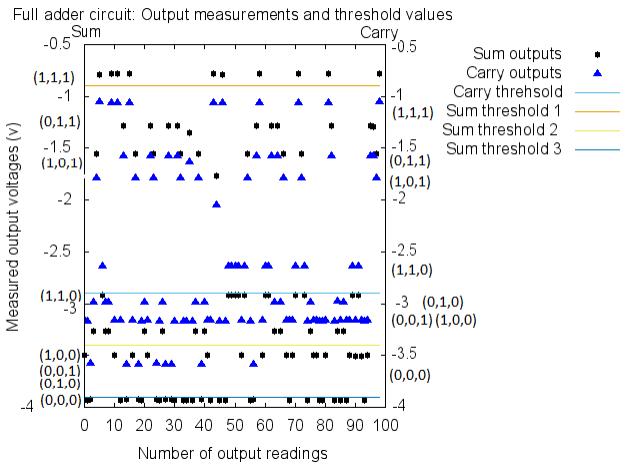


Figure 9: Material response for the full-adder circuit; output measurements and thresholds.

evolvable material. For the experiments reported here, material with 0.1% SWCNTs concentration was used. In future work, the material with varied concentrations will be tested for similar computation problems.

Acknowledgments

The research is supported by European's Community Seventh Framework programme (FP7/2007-2013) under the grant agreement No. 317662 (NAno Scale Engineering for Novel Computation using Evolution - NASCENCE (http://www.nascence.eu)). We are thankful to Odd Rune Lykkebø and Gunnar Tufte from the Norwegian University of Science and Technology for the Mecobo platform.

References

[1] S. Harding and J. F. Miller, "Evolution in materio," in *Encyclopedia of Complexity and Systems Science*, R. A. Meyers, Ed. Springer New York, 2009, pp. 3220–3233. [Online]. Available: http://dx.doi.org/10.1007/978-0-387-30440-3_190

[2] A. Thompson, "An evolved circuit, intrinsic in silicon, entwined with physics," in *Evolvable Systems: From Biology to Hardware*. Springer, 1997, pp. 390–405.

[3] S. Harding, J. F. Miller, and E. A. Rietman, "Evolution in materio: Exploiting the physics of materials for computation," *arXiv preprint cond-mat/0611462*, 2006.

[4] A. Kotsialos, M. K. Massey, F. Qaiser, D. A. Zeze, C. Pearson, and M. C. Petty, "Logic gate and circuit training on randomly dispersed carbon nanotubes." *International Journal of Unconventional Computing.*, vol. 10, no. 5-6, pp. 473–497, September 2014.

[5] M. K. Massey, A. Kotsialos, F. Qaiser, D. A. Zeze, C. Pearson, D. Volpati, L. Bowen, and M. C. Petty, "Computing with carbon nanotubes: Optimization of threshold logic gates using disordered nanotube/polymer composites," *Journal of Applied Physics*, vol. 117, no. 13, p. 134903, 2015.

[6] D. Volpati, M. K. Massey, D. Johnson, A. Kotsialos, F. Qaiser, C. Pearson, K. Coleman, G. Tiburzi, D. A. Zeze, and M. C. Petty, "Exploring the alignment of carbon nanotubes dispersed in a liquid crystal matrix using coplanar electrodes," *Journal of Applied Physics*, vol. 117, no. 12, p. 125303, 2015.

[7] M. K. Massey, D. Volpati, F. Qaiser, A. Kotsialos, C. Pearson, D. A. Zeze, and M. C. Petty, "Alignment of liquid crystal/carbon nanotube dispersions for application in unconventional computing," in *AIP Conference Proceedings - ICNAAM*, vol. 1648. AIP, 2015, pp. 280 009–1.

[8] O. R. Lykkebø, S. Harding, G. Tufte, and J. F. Miller, "Mecobo: A hardware and software platform for in materio evolution," in *Unconventional Computation and Natural Computation*, ser. Lecture Notes in Computer Science, O. H. Ibarra, L. Kari, and S. Kopecki, Eds. Springer International Publishing, 2014, vol. 8553, pp. 267–279. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-08123-6_22

[9] S. Harding, J. Neil, K.-P. Zauner, J. F. Miller, and K. Clegg, "A framework for the automatic identification and extraction of computation from materials," University of York, Tech. Rep., 2006.

[10] S. Tsuda, K. P. Zauner, and Y.-P. Gunji, "Robot control with biological cells," *Biosystems*, vol. 87, no. 2, pp. 215–223, 2007.

[11] J. Jones, J. Whiting, and A. Adamatzky, "Quantitative transformation for implementation of adder circuits in physical systems," *Biosystems*, vol. 134, pp. 16 – 23, 2015.

[12] M. Amos, D. A. Hodgson, and A. Gibbons, "Bacterial self-organisation and computation," 2005.

[13] S. Prasad, M. Yang, X. Zhang, C. Ozkan, and M. Ozkan, "Electric field assisted patterning of neuronal networks for the study of brain functions," *Biomedical Microdevices*, vol. 5, no. 2, pp. 125–137, 2003.

[14] S. Harding and J. F. Miller, "Evolution in materio: a tone discriminator in liquid crystal," in *Evolutionary Computation, 2004. CEC2004. Congress on*, vol. 2, 2004, pp. 1800–1807 Vol.2.

[15] S. Bose, C. Lawrence, Z. Liu, K. Makarenko, R. van Damme, H. Broersma, and W. van der Wiel, "Evolution of a designless nanoparticle network into reconfigurable boolean logic," *Nature nanotechnology*, 2015.

[16] J. F. Miller and M. Mohid, "Function optimization using cartesian genetic programming," in *Proceedings of the 15th annual conference companion on Genetic and evolutionary computation*. ACM, 2013, pp. 147–148.

[17] M. Mohid, J. F. Miller, S. Harding, G. Tufte, O. R. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving machine learning classification problems using materials," in *Parallel Problem Solving from Nature–PPSN XIII*. Springer, 2014, pp. 721–730.

[18] M. Mohid, J. F. Miller, S. Harding, G. Tufte, O. R. Lykkebo, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving bin packing problems using materials," in *Evolvable Systems (ICES), 2014 IEEE International Conference on*. IEEE, 2014, pp. 38–45.

[19] ——, "Evolution-in-materio: A frequency classifier using materials," in *Evolvable Systems (ICES), 2014 IEEE International Conference on*. IEEE, 2014, pp. 38–45.

[20] NASCENCE project (ICT 317662), "Report on suitable computational tasks of various difficulties," 2013, deliverable D4.2.

[21] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the sixth international symposium on micro machine and human science*, vol. 1. New York, NY, 1995, pp. 39–43.

[22] X. Hu, R. C. Eberhart, and Y. Shi, "Swarm intelligence for permutation optimization: a case study of n-queens problem," in *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*. IEEE, 2003, pp. 243–246.

[23] R. S. Verma, V. Singh, and S. Kumar, "Dna sequence assembly using particle swarm optimization," *International Journal of Computer Applications*, vol. 28, no. 10, 2011.