

## Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories

Dmitri B. Strukov and Konstantin K. Likharev\*

Stony Brook University, Stony Brook, NY 11794-3800, U.S.A.

Phone: 631-632-8159

Fax: 631-632-4977

\*E-mail: [klikharev@notes.cc.sunysb.edu](mailto:klikharev@notes.cc.sunysb.edu)

### *Abstract*

We have calculated the maximum useful bit density that may be achieved by the synergy of bad bit exclusion and advanced (BCH) error correcting codes in prospective crossbar nanoelectronic memories, as a function of defective memory cell fraction. While our calculations are based on a particular (“CMOL”) memory topology, with naturally segmented nanowires and an area-distributed nano/CMOS interface, for realistic parameters our results are also applicable to “global” crossbar memories with peripheral interfaces. The results indicate that the crossbar memories with a nano/CMOS pitch ratio close to 1/3 (which is typical for the current, initial stage of the nanoelectronics development), may overcome purely semiconductor memories in useful bit density if the fraction of nanodevice defects (stuck-on-faults) is below ~15%, even under rather tough, 30 ns upper bound on the total access time. Moreover, as the technology matures, and the pitch ratio approaches an order of magnitude, the crossbar memories may be far superior to the densest semiconductor memories by providing, e. g., a 1 Tbit/cm<sup>2</sup> density even for a plausible defect fraction of 2%. These highly encouraging results are much better than those reported in literature earlier, including our own early work, mostly due to more advanced error correcting codes.

## 1. INTRODUCTION

The performance of many electronic products, including all computing systems (from palmtops to supercomputers) and many portable consumer electronic devices (cell phones, portable digital video and music players, digital cameras, etc.), substantially depends on memory.<sup>1-3</sup> Actually, the very success of some electronic systems in the future may depend on the memory scaling prospects.<sup>4</sup> However, the conventional semiconductor memories (including SRAM, DRAM and Flash) are already approaching their scaling limits.<sup>5</sup> This is why several prospective memories, including SONOS, FRAM, MRAM, and PCM (see, for example, the “Emerging Research Devices” section of the International Technology Roadmap for Semiconductors<sup>5</sup>) have been suggested and are actively explored. Unfortunately, the electronic industry has not yet fully recognized the potential value of “crossbar” (a.k.a. “resistive”, or “1R” or “0T”) memories whose cell does not require a transistor and hence may have the smallest area.

In such memories, information bits are stored in a two-terminal bistable nanodevices (called “latching switches” or “programmable diodes”) which are formed at each crosspoint of nanowire crossbar structures (Fig. 1a). The device  $I$ - $V$  curve (Fig. 1b) has two branches corresponding to its two possible internal states. (In the equivalent circuits shown in Figs. 1c, d, this bistability is represented by a switch. Note that a quantitative analysis of memory performance, like the one carried out in this work, requires more accurate equivalent circuits – see Sec. 6 below.) In the low-resistive state presenting binary 1, the nanodevice is essentially a diode, so that the application of voltage  $V_t < V_{\text{READ}} < V_+$  to one (say, horizontal) nanowire leading to the memory cell gives substantial current injection into the second wire (Fig. 1c). This current pulls up voltage  $V_{\text{out}}$  which can now be read out by a sense amplifier. (The diode’s property to have low

current at voltages above  $-V_t$  prevents parasitic currents which might be induced in other state-1 cells by the output voltage – see the red line in Fig. 1c and a quantitative analysis in Ref. 6. It also reduces shot noise in the output line – see Sec. 6 for details.)

In state 0 (which presents binary zero) the crosspoint current is very small, giving a nominally negligible contribution to output signals at readout. In order to switch it to state 1 (i.e., write binary 1 into the cell), the two nanowires leading to the device are fed by voltages  $\pm V_{\text{WRITE}}$  (Fig. 1d), with  $V_{\text{WRITE}} < V_+ < 2V_{\text{WRITE}}$ . (The left inequality ensures that this operation does not disturb the state of “semi-selected” devices contacting just one of the biased nanowires.) The write 0 operation is performed similarly using the reciprocal switching with threshold  $V$ . (Fig. 1b). It is evident from Figs. 1c, d that the read and write operations may be performed simultaneously with all cells of one row.<sup>7</sup>

Two-terminal devices with the functionality of programmable diodes (Fig. 1b) have been demonstrated using several structures, notably including amorphous metal-oxide films,<sup>8-15</sup> relatively thick organic films with<sup>16-20</sup> and without<sup>21,22</sup> embedded metallic clusters, self-assembled molecular monolayers,<sup>23-27</sup> and thin chalcogenide layers.<sup>28</sup> A significant advantage of the crossbar memories over other prospective memory technologies is that such two-terminal nanodevices with necessary characteristics have only one (in Fig. 1a, vertical) critical dimension. This dimension may be defined by the thickness of deposited film(s) or self-assembled molecular monolayers and as a result may be scaled down, sustaining very high precision, well below 10 nm without an unacceptable increase of fabrication costs.

The second important advantage of crossbar memories is their small cell footprint  $(2F_{\text{nano}})^2$ , where  $F_{\text{nano}}$  is the nanowire half-pitch. This fact is especially significant because arrays of parallel nanowires may be fabricated by several advanced patterning technologies (such as

nanoimprint<sup>29-31</sup> or interference lithography<sup>32,33</sup>) which may combine acceptable fabrication speed with very small  $F_{\text{nano}}$ . Indeed, nanoimprint technology has already allowed a crossbar memory prototype with  $F_{\text{nano}} = 30 \text{ nm}$ <sup>27</sup> and a nanowire crossbar with  $F_{\text{nano}} = 17 \text{ nm}$ <sup>34</sup> to be experimentally demonstrated, and there are good prospects for the half-pitch reduction to 3 nm or so within the next decade.<sup>29-31</sup>

The last number corresponds to an unprecedented memory density in excess of 1 Terabit/cm<sup>2</sup>, i. e. three orders of magnitude higher than that in existing semiconductor memory chips.<sup>35</sup> On the other hand, the relatively high number of defective crosspoint nanodevices may be an obstacle for the implementation of this advantage, especially if improper architectures are used. This is why crossbar memories deserve a thorough theoretical analysis.

The goal of this work has been to extend and improve significantly the first results obtained in this direction.<sup>36,37</sup> In the next Section 2 we describe the most important issues which have to be addressed at crossbar memory analysis, as well as the results and shortcomings of the initial efforts. Sections 3-6 are devoted to a detailed description of the considered memory architecture and the assumptions used for our present analysis, while its results are described and discussed in Sec. 7 and 8.

## 2. CROSSBAR MEMORY ISSUES

In order to be relevant, any analysis of crossbar memories has to take into account two main challenges faced by their practical development.

(i) The high-resolution patterning technologies mentioned above do not offer equally accurate layer alignment (“overlay”). This is why interfacing nanowire crossbars to peripheral CMOS circuits (fabricated using cruder patterning technologies with half-pitch  $F_{\text{CMOS}} \gg F_{\text{nano}}$ )

presents a challenge. Several initial suggestions for forming such interfaces at the crossbar array periphery (for their reviews see Refs. 38-41) do not seem very practicable.<sup>42</sup> Recently, we suggested<sup>42-45</sup> a new approach (dubbed “CMOL”) in which the interface is provided all over the chip surface, using pins with the CMOS pitch but nanoscale-sharp tips (Fig. 2). The main feature of the CMOL topology is that it provides a unique access from the CMOS subsystem to each crossbar nanowire (whether in the bottom or the top layer) with the theoretical 100% fabrication yield even in the absence of *any* alignment between the CMOS and crossbar subsystems. (Note, that the last statement is only true for the recently introduced<sup>45</sup> CMOL version in which each pin, going to the upper nanowire level, intentionally interrupts a lower level wire – see Fig. 2.) Moreover, the “red” pins reaching the lower level of the crossbar provide such access to each segment of the bottom level nanowires. (Such segments are naturally formed by the breaks provided by the “blue” pins going to the upper level.) In the architecture considered in this work, we take the CMOL wire segmentation into account, but for realistic parameters all our results are also valid for “global” crossbar memories with peripheral interfaces.

(ii) For some programmable diode types, the device bistability mechanism is not yet clear, but for the currently most reproducible metal-oxide devices<sup>15</sup> it is probably due to electron trapping in localized states.<sup>46</sup> The basic drawback of the diodes based on this mechanism is that in order to feature reasonable current density, the distance between active localized states cannot be much smaller than  $\sim 3$  nm.<sup>47,48</sup> In order to be statistically reproducible, the device should have a large number of the states. This is why the extension of the excellent reproducibility demonstrated for crosspoint devices with  $F_{\text{nano}} > 100 \text{ nm}^2$  (as in Ref. 15) to cells with  $F_{\text{nano}} < 10 \text{ nm}^2$  may present a major challenge.

This problem may be addressed using uniform self-assembled monolayers of specially designed molecules<sup>42,49</sup> implementing single-electron latching switches.<sup>50</sup> (Metal-based, low-temperature prototypes of such switches, with multi-hour retention times, have been demonstrated experimentally.<sup>51,52</sup> However, so far molecular implementations have been only demonstrated<sup>53-57</sup> for the main components of these devices, single-electron transistors.) A major challenge on this way is the reproducibility of the interface between the monolayer and the second (top) metallic electrode, because of the trend of the metallic atoms to diffuse inside the monolayer during the electrode deposition.<sup>55</sup> Recent very encouraging results towards the solution of this problem have been obtained using an intermediate layer of a conducting polymer.<sup>58</sup> Another opportunity is to terminate the synthesized molecules, before their self-assembly, with special metallic clusters or large acceptor groups which would play the role of “floating electrodes”.<sup>42</sup> The recently reported experimental results<sup>59</sup> may be considered as the first step toward the practical implementation of this opportunity.

It is natural to expect that at the initial stage of development of all these devices, their fabrication yield for  $F_{\text{nano}} < 30$  nm will be considerably below 100%, and, for  $F_{\text{nano}} \sim 3$  nm, will possibly never approach this limit closer than a few percent. This is why the crossbar memory architectures have to ensure high defect tolerance. The two main approaches for fighting errors in memories are reconfiguration, i.e. the replacement of bad memory cells with spare ones, and error correcting codes (ECC).<sup>2,60</sup> For higher defect rates the best defect tolerance can be achieved by combining these two techniques.<sup>61</sup>

Earlier we investigated the density vs. defect tolerance tradeoff in crossbar memories, provided by such synergy, using only very light (Hamming) codes.<sup>36</sup> The results were not too optimistic: with a reasonably fast “Repair Most” algorithm of bad line replacement, an order-of-

magnitude advantage over best possible semiconductor memories<sup>62</sup> might be achieved for the fraction of bad memory cells not exceeding  $\sim 0.1\%$ .

DeHon *et al.*<sup>37</sup> and Sun *et al.*<sup>63</sup> have indicated that much better defect tolerance (up to  $\sim 10\%$ ) may be achieved using more advanced ECC, e.g., Reed-Solomon and Bose-Chaudhuri-Hocquenghem (BCH) codes.<sup>64</sup> Unfortunately, in those works, the contributions of the circuits implementing these codes to the memory access time (which for some codes may be extremely large) and the total memory area have not been estimated. Also, the account of the finite leakage current through nominally closed crosspoints (which has been neglected in Ref. 37) may change the memory scaling rather substantially.<sup>6, 65</sup>

In this work we present a detailed analysis of possible tradeoff between density, defect tolerance, and speed performance of crossbar memories on the example of their CMOL variety. We reach high defect tolerance via synergy of bad bit exclusion (using the optimized granularity) with advanced BCH error correcting codes<sup>64</sup> with optimized parameters.

### 3. MEMORY ARCHITECTURE AND OPERATION

We consider a natural top structure of CMOL memory (Fig. 3a), which is essentially similar to that accepted in Ref. 36. It is a rectangular array of  $L$  crossbar memory banks (“blocks”) – Fig. 3a. During a single operation, a particular row of CMOL blocks is accessed with the help of block address decoders. The block architecture (Fig. 3b) is specific for the CMOL interface which allows the placement of CMOS “relay” cells under the nanowire crossbar. These cells are controlled by CMOS-level decoders, four per each block (Fig. 3b). At each elementary operation, one pair of block decoders (shown in magenta in Fig. 3b, as well as Figs. 4 and 5 below) addresses one vertical and one horizontal CMOS line, and thus selects a certain relay cell at their

crosspoint. This cell (Fig. 5) applies the data signal to a “red” interface pin contacting a bottom-layer nanowire. The other pair of decoders (shown in violet in Figs. 3b, 4 and 5) selects a set of different relay cells which provide similar biasing of the corresponding top level nanowires through “blue” pins. These nanowires may now address all crosspoint nanodevices (memory cells) of a particular nanowire segment. Thus the four decoders of the block, working together, can provide every memory cell of the segment with voltages necessary for the read and write operations.

The remaining circuitry shown in Fig. 3b, i.e. CMOS-based mapping table and address control circuitry, is needed to convert the logical (external) addresses, which are fed to the CMOL blocks, into internal addresses of memory cells inside the block. In particular, the mapping table converts the logical address of the segment (which is the same for all selected blocks) into a pair of block-specific physical addresses,  $A_{\text{col1}}$  and  $A_{\text{row1}}$ , and CMOS-implemented decoders activate the corresponding CMOS-level lines.

Figure 4 shows the low-level structure of the CMOL memory for a particular value of the main topological parameter of CMOL,  $r = 4$ . (For realistic parameters,  $r \gg 1$  – see Sec. 7 and 8) The top-level nanowires (here shown quasi-horizontal) stretch over the whole block, but the low-level (nearly-vertical) nanowires are naturally cut into segments of equal length. An elementary analysis of the CMOL geometry (Fig. 2b) shows that each nanowire segment stretches over  $r$  CMOS cells and contacts  $r^2$  (in Fig. 4, sixteen) crosspoint nanodevices.

Signals  $A_{\text{col1}}$  and  $A_{\text{row1}}$  are applied to CMOS wires, feeding the “red” lines of the corresponding CMOS-implemented relay cells (Fig. 5). By opening all pass transistors of the row,  $A_{\text{row1}}$  selects a specific “red” pin of column  $A_{\text{col1}}$ , so that the data  $A_{\text{col1}}$  are fed only to a specific nanowire segment contacting  $r^2$  crosspoint nanodevices. In parallel, addresses  $A_{\text{col1}}$  and

$A_{\text{row}1}$  are fed to the CMOS-based address control circuitry to generate another pair of physical addresses  $A_{\text{row}2}$  and  $A_{\text{col}2}$ . Signal  $A_{\text{row}2}$  opens the “blue”-pin pass transistors in relay cells of a row, and therefore connects each of  $r^2$  quasi-horizontal nanowires of the top layer to a specific CMOS lines (shown purple), thus enabling a read or write operation.

For large CMOL arrays (say, a square array with  $W$  relay cells on a side, with  $W \gg r^2$ ), most nanowire segments are well inside the array. In order to address such segments,  $A_{\text{row}2}$  might just reproduce  $A_{\text{row}1}$ . The problem with such scheme is that it would allow addressing only  $W(W - r^2)$  internal nanowire segments, of the total number  $W^2$ . In order to decrease the associated loss of  $Wr^4$  memory cells (of the total  $W^2r^2$  relay cells in the block), we prefer to address simultaneously two lines:

$$A_{\text{row}2a} = A_{\text{row}1} + r/2, \quad (1)$$

$$A_{\text{row}2b} = A_{\text{row}1} - r/2. \quad (2)$$

Such addressing scheme (Fig. 4) reduces the loss to just  $Wr^3$  memory cells per block. (For the case shown in Fig. 4, the lost cells are located in two rectangular  $W \times r/2$  areas on the top and at the bottom of the array.)<sup>66</sup> The associated area penalty is negligible in most cases (see the analysis below).

Note that of  $W$  output CMOS lines (purple arrows on the bottom of each panel of Fig. 4), at each operation only  $r^2$  lines are connected to nanodevices of the selected fragment. The selection of these useful lines (or a part of them, see Sec. 4 below) and their connection to the system output are provided by the data decoder controlled by signal  $A_{\text{col}2}$  (Fig. 3). The necessary circuit is rather simple (essentially, a barrel shifter) and may be readily implemented in the CMOS subsystem.

#### 4. DEFECT TOLERANCE CALCULATION

We have calculated the tolerance of our memories to “hard” (fabrication-induced) defects, using the following assumptions:

- (i) defective nanodevice cells are randomly distributed, with probability  $q$ , around the memory;<sup>67</sup>
- (ii) the effect of defective interface pins, nanowires, and CMOS components is negligibly small; and
- (iii) the defects correspond to permanently disconnected crosspoints (i.e. are similar to “stuck-on-open” faults).

These assumptions seem reasonable at least at the present stage of development of molecular electronics, and have been made in most analyses of nanoelectronic circuits.<sup>36,37, 45, 68-71</sup> (In future, it is certainly desirable to extend the analysis to “stuck-on-close” defects and unintentional nanowire breaks. In this paper, we will only briefly discuss the possible effect of such defects on the performance of the memory.)

In the synergetic approach of combining the memory array reconfiguration with ECC,<sup>61</sup> memory cells are divided into fragments of certain size (“granularity”). Each of these fragments is tested using ECC circuitry, and those of them which may not be ECC-corrected are excluded from operation. (For that, the addresses of good fragments are written into the mapping table, see Fig. 3). If the fraction  $q$  of bad bits is large, the large granularity of exclusion is impracticable, due to the exponential growth of the number of necessary redundant resources. (In particular, this was the reason of the relatively poor defect tolerance achieved in our previous work,<sup>36</sup> where only global nanowires might be excluded.) On the other hand, fine granularity requires an unacceptably large mapping table. In this work we use a new, more flexible approach when the granularity of exclusion is not related to the physical structure of the memory array. This means

that the data fragment length, equal to  $g$  nanowire segments (i.e.  $gr^2$  memory cells) may be either smaller or larger than the one segment (which has  $r^2$  memory cells).

In the former case ( $g < 1$ ), the fragment is physically placed on a part of one nanowire segment. This can be easily implemented by keeping an additional “displacement” address inside the mapping table, and an additional circuitry which multiplies the displacement by  $gr^2$  and adds the result to  $A_{\text{col2}}$ . In the latter case ( $g > 1$ ), the fragment is physically placed (at the same intrablock address) into  $g$  nanowire segments of  $g$  adjacent blocks of the same block row – see Fig. 3a. (Their location inside the same block would result in a larger block size, and hence in larger global nanowire capacitance and delay time – see Sec. 6.) This requires the address mapping table to be shared between  $g$  neighboring blocks.

For a binary ECC with length  $n$  and the information length  $k$ , which can fix up to  $t$  errors in any of the  $n$  bits, the probability  $P_f$  to fix a fragment is

$$P_f = \left[ \sum_{i=0}^t \binom{n}{i} q^i (1-q)^{n-i} \right]^{gr^2/n}. \quad (3)$$

Here  $n$  is assumed to divide  $gr^2$  exactly. (If this is not so, the bits in the remainder cannot be used and would be wasted.) If each block has  $a$  spare and  $m$  useful fragments (with the sum  $m + a = W^2$ ), the probability  $P_{\text{sb}}$  for a “superblock” (consisting of  $g$  adjacent blocks which share data fragments) to be fully functional can be calculated as

$$P_{\text{sb}} = \sum_{i=0}^a \binom{m+a}{i} (1-P_f)^i P_f^{m+a-i}, \quad (4)$$

and the yield  $Y$  of the total memory is

$$Y = P_{\text{sb}}^{L/g}. \quad (5)$$

In order to characterize the defect tolerance, we fix  $Y$  at a certain level (usually, 90%), and numerically optimize the granularity (fragment length  $g$ ) and ECC parameters  $n$  and  $k$  to

calculate the maximum manageable fraction  $q$  of bad memory cells. Since for realistic parameters (in particular,  $W^2 \gg 1$ ),  $P_f$  drops from nearly one to zero extremely fast at certain  $q \ll 1$ , the block size (and hence the number  $L$  of blocks at a fixed total memory size  $LW^2r^2$ ) and  $Y$  virtually do not affect these results.

## 5. AREA CALCULATION

### 5.1 Total Area and Capacity

Since for realistic block dimensions the area of block decoders (Fig. 3a) is negligible,<sup>36</sup> the total memory area may be calculated as

$$A = L \times A_{\text{block}} = L \times (A_{\text{array}} + A_{\text{cell decoder}} + A_{\text{drive/sense}} + g^{-1}A_{\text{control}} + g^{-1}A_{\text{mapping table}}), \quad (6)$$

while useful capacity of the memory is

$$N = \frac{L}{g} mk \left[ \frac{gr^2}{n} \right], \quad (7)$$

where  $m$  is the number of good data fragments per block,  $L/g$  is the number of “superblocks”, and  $gr^2$  the number of memory cells in one fragment - see Sec. 4 above. The ratio  $k/n < 1$  reflects the area loss due to the ECC. The memory density may be characterized by the total area per one useful cell, i.e. the ratio  $A/N$ , which may be conveniently expressed in the units of  $(F_{\text{CMOS}})^2$ .

### 5.2. Crossbar Arrays

The area of a CMOL array of  $W \times W$  relay cells is simply

$$A_{\text{array}} = W^2 \times (2\beta F_{\text{CMOS}})^2, \quad (8)$$

where  $\beta \geq 1.6$ , since the minimum area of the CMOS cell housing two minimum-width pass transistors (Fig. 5) may be estimated as  $10(F_{\text{CMOS}})^2$  - see, e.g. Ref. 36.

### 5.3. Decoders

Our decoders are essentially the pass-gate-based multiplexers (Fig. 6a), with all CMOS transistors of minimum width.<sup>36</sup> We assume that one pass gate, consisting of one nMOS and one pMOS transistors, can be placed in the area of  $4F_{\text{CMOS}} \times 2\beta F_{\text{CMOS}}$ , i.e. matching the CMOS pitch of the memory array. In this case the height of the  $2^W$ -input decoder is simply  $2\log_2 W \times 2F_{\text{CMOS}}$ . Similarly, we have assumed that the height of the  $r^2$ -out-of- $W$  barrel shifter implementation (Fig. 6b) is equal to  $(2\log_2 W + r^2) \times 2F_{\text{CMOS}}$ .

Considering a 6-transistor latch-style implementation of sense amplifiers<sup>3</sup> with  $A_{\text{sense}} = 100 \times (F_{\text{CMOS}})^2$ , and 2-transistor drive buffers (inverters) with  $A_{\text{drive}} = 25 \times (F_{\text{CMOS}})^2$ , the increase in the linear size of the crossbar array in the horizontal direction, due to decoders, is

$$\Delta w_1 = (12\log_2 W + 2A_{\text{drive}}/2\beta F_{\text{CMOS}}) \times F_{\text{CMOS}}. \quad (9)$$

Similarly, the array increase in the vertical dimension due to one decoder and one barrel shifter is

$$\Delta w_2 = (8\log_2 W + 2r^2 + (2A_{\text{drive}} + A_{\text{sense}})/2\beta F_{\text{CMOS}}) \times F_{\text{CMOS}}. \quad (10)$$

#### 5.4. Control Circuitry

The control circuitry consists of two adders of length  $\log_2 W$  (bits). Since it is relatively short, we assume the ripple carry implementation with the area of a 1-bit full adder equal to  $2000 \times (F_{\text{CMOS}})^2$ ,<sup>72</sup> so that

$$A_{\text{control}} = 4000 \log_2 W \times (F_{\text{CMOS}})^2. \quad (11)$$

Control circuitry may be placed into the corner areas between the adjacent cell decoders; therefore the area contribution from the first four terms in Eq. (6) may be calculated as

$$\begin{aligned} & A_{\text{array}} + A_{\text{cell decoder}} + A_{\text{drive/sense}} + g^{-1} A_{\text{control}} \\ &= (W \times 2\beta F_{\text{CMOS}} + \Delta w_1)(W \times 2\beta F_{\text{CMOS}} + \Delta w_2) + \max(0, g^{-1} A_{\text{control}} - \Delta w_1 \Delta w_2). \end{aligned} \quad (12)$$

The area of mapping table implemented in CMOS is assumed to be

$$A_{\text{mapping table}} = 2m \log_2 W \times (2F_{\text{CMOS}})^2. \quad (13)$$

## 5.5. ECC Decoders

We have studied the area and delay tradeoffs of fast bit-parallel decoding for a popular class of multiple-bit error-correcting linear codes – BCH codes. (For example, the Reed-Solomon codes<sup>64</sup> which are a subclass of the BCH family, are broadly used in today’s flash memories and storage devices.<sup>2</sup>) While BCH ECC may not be the absolutely best option for our particular problem,<sup>73</sup> they are among the most efficient short codes (with  $n \leq 512$ ), and certainly much more powerful than the Hamming codes used in Ref. 36. Our model of a fast bit-parallel decoder for BCH codes is presented Appendix. Its analysis shows, in particular, that the decoder area, in the units of  $(F_{\text{CMOS}})^2$ , may be estimated as

$$A_{\text{BCH}} \approx 125nt(\log_2 n)^2 + 40n(\log_2 n)^2 + 250nt \log_2 n + 190n \log_2 n + 1200t(\log_2 n)^2 + 300t^2 \log_2 n. \quad (14)$$

Figure 7 shows several examples of application of this formula. The results indicate that the decoder area (even for this completely parallel, i. e., the most spacious, implementation) is negligible in comparison with that of memory arrays, so that it will not be considered in the optimization procedure described below.

## 6. SPEED AND POWER

### 6.1. ECC Decoder

The BCH ECC decoder delay calculation is also described in the Appendix. In the units of the standard CMOS fan-out-of-four delay, it may be expressed as

$$\tau_{\text{BCH}} \approx 0.7t \log_2 n + 0.7t + 8t \log_2 \log_2 n + 3.6t \log_2 t + 2.5 \log_2 \log_2 n + 1.8 \log_2 t + 1.8. \quad (15)$$

Figure 7 shows several results obtained using this formula. We will now show that the delays of read and write operations<sup>74</sup> in the crossbar arrays with metallic nanowires, as well as those of

moving data between the blocks, are much shorter than that of ECC decoding, expressed by Eq. (15). Indeed, let us first estimate the delays inside the blocks.

### 6.2. Nanowires

The resistance of metallic nanowires may be calculated<sup>70</sup> as that of conductors with cross-section  $F_{\text{nano}} \times F_{\text{nano}}$  and resistivity  $\rho$  adjusted to diffusive surface scattering of electrons:

$$\rho \approx \rho_0 \times (1 + \lambda / F_{\text{nano}}), \quad (16)$$

where  $\rho_0$  is the table (bulk) resistivity of a pure metal. We have assumed values  $\rho_0 = 2 \mu\Omega\text{-cm}$  and  $\lambda = 10 \text{ nm}$  which are typical for good metals at room temperature. The capacitance of nanowires per unit length for the considered range of  $F_{\text{nano}}$  is about  $0.2 \text{ fF}/\mu\text{m}$ .<sup>70</sup>

### 6.3. Crosspoint Nanodevices

We have assumed that each crosspoint programmable diode is implemented as a parallel connection of  $D$  single-electron latching switches,<sup>50-52</sup> so that the resistance of the single crosspoint nanodevice is  $R_{\text{ON}}/D$  and  $R_{\text{OFF}}/D$  in the ON and OFF states, correspondingly. (Estimates for the devices based on electron trapping in random localized states are in the same ballpark, because the basic physics of their operation<sup>46</sup> is very close to that of single-electron devices.) Based on the experimental data for self-assembled monolayers (see, e.g., Ref. 76), the footprint of a single molecule may be estimated as  $0.25 \text{ nm}^2$ ; so for  $D$  we have used the following value:

$$D \approx (F_{\text{nano}})^2 / (0.25 \text{ nm}^2). \quad (17)$$

Resistances  $R_{\text{ON}}$  and  $R_{\text{OFF}}$  are related by the equation for the second-order quantum effect, elastic co-tunneling:<sup>77</sup>

$$\frac{R_{\text{OFF}}}{R_{\text{ON}}} = \frac{R_{\text{ON}}}{R_{\text{Q}}}, \quad (18)$$

where  $R_Q \equiv \hbar/e^2 \approx 4.1 \text{ k}\Omega$  is the quantum unit of resistance. (The thermally-induced suppression of  $R_{\text{OFF}}$  is typically negligible.<sup>36</sup>)

#### 6.4. Intra-block Delay

The necessary condition for a reliable read operation is that the voltage swing between the worst cases of reading state “1” and state “0” on the input of the sense amplifier is approximately 10 times larger than the r.m.s. voltage fluctuations  $\Delta V_N$  due to the noise. (Here, the factor 10 results from the assumed Gaussian distribution of noise and the requirement of keeping the bit error rate below  $10^{-23}$ . This is sufficient, e.g., to provide a  $10^8$ -hour mean time to failure at the very aggressive aggregate memory bandwidth of 1 Tbit/s. Note that the factor 23 accepted in Ref. 70 was excessive, leading to a certain underestimate of the possible performance of CMOL FPGA logic circuits.) The voltage swing can be found from the equivalent circuit shown in Fig. 8. Here we have assumed that all unselected nanowire segments are pulled to the ground through resistance  $(R_{\text{wire}}R_{\text{OFF}}/D)^{1/2}$  which is the input resistance of a semi-infinite ladder formed by the wire resistances  $R_{\text{wire}}$  of the length  $2F_{\text{nano}}$  and crosspoint nanodevices in OFF state  $R_{\text{OFF}}/D$ . Note that the pulldown resistance  $R_{\text{pd}}$  shown in Fig. 5 may be neglected here since it is much less than  $R_{\text{OFF}}/D$ . Also, in the equivalent circuit we neglect the resistance of bottom layer nanowires, which is at most  $1/2r^2R_{\text{wire}}$ , i.e. much smaller than the worst-case resistance  $r^3R_{\text{wire}}$  of the top level (quasi-horizontal) nanowire.

For all reasonable parameter sets, it is possible to pick a realistic value of  $R_{\text{ON}}$  such that the following conditions are satisfied:

$$R_{\text{ON}}/D \ll r^3R_{\text{wire}}, \quad (19)$$

$$R_{\text{sense}} \ll r^3R_{\text{wire}}, \quad (20)$$

$$R_{\text{sense}} \ll (R_{\text{wire}}R_{\text{OFF}}/D)^{1/2}. \quad (21)$$

In this case, all the formulas describing the equivalent circuit may be significantly simplified. In particular, the voltage swing is simply

$$V_{\text{swing}} \approx \frac{R_{\text{sense}}}{r^3 R_{\text{wire}}} V_{\text{READ}}. \quad (22)$$

In general, the equivalent circuit shown in Fig. 8 allows one to calculate the total noise  $\Delta V_N$  contributed by the shot noise on the nanodevices in their ON and OFF states, and the thermal (Johnson-Nyquist) noise on the nanowires and  $R_{\text{sense}}$ . However, the inequalities (19)-(21) ensure that  $\Delta V_N$  is dominated by the thermal noise of  $R_{\text{sense}}$ , so that

$$\Delta V_N \approx [k_B T / (2C_{\text{wire}} + C_{\text{CMOS}})]^{1/2}, \quad (23)$$

where  $C_{\text{wire}}$  is the capacitance of the global quasi-horizontal nanowire, of length  $Wr \times 2F_{\text{nano}}$ , while  $C_{\text{CMOS}}$  is a capacitance of the CMOS data line of the same length (see the violet data lines in Fig. 4). The typical value of  $C_{\text{CMOS}}$  for the considered CMOS technology nodes is about 0.1 fF/ $\mu\text{m}$ .<sup>5</sup> The doubling of  $C_{\text{wire}}$  in Eq. (23) is due to the fact that two CMOS select lines have to be activated for a read operation (Fig. 4) and therefore two quasi horizontal nanowire lines will be charged.

The full intrablock latency may be estimated as

$$\tau_{\text{intrablock}} \approx R_{\text{sense}} (2C_{\text{wire}} + C_{\text{CMOS}}). \quad (24)$$

### 6.5. Interblock Delay

This delay can be reduced by adding repeaters on the periphery of each block, so that the delay is

$$\tau_{\text{interblock}} \approx 1/2 \sqrt{LR_{\text{CMOS}} C_{\text{CMOS}}}, \quad (25)$$

where resistance  $R_{\text{CMOS}}$  is, similarly to  $C_{\text{CMOS}}$ , proportional to the linear block size. Here we assume that the area overhead due to the repeaters is insignificant.

## 6.6. Example

Now let us consider a particular, but typical case  $W = 256$ ,  $F_{\text{CMOS}} = 45$  nm and  $F_{\text{nano}} = 4.5$  nm (and hence  $r = 16$ ). In this case the nanowire resistance  $R_{\text{wire}}$  turns out to be about  $14 \Omega$ ,  $r^3 R_{\text{wire}} \approx 57 \text{ K}\Omega$ , while  $C_{\text{wire}} = 7.4$  fF. By using nanodevices with  $R_{\text{ON}} = 400 \text{ K}\Omega \cong 10^2 R_Q$ , and therefore  $R_{\text{OFF}} \cong 40 \text{ G}\Omega$ , the resistances of the crosspoint devices are, respectively,  $R_{\text{ON}}/D \approx 5 \text{ K}\Omega$ , and  $R_{\text{OFF}}/D \approx 0.5 \text{ G}\Omega$ , since for the considered value of  $F_{\text{nano}}$  the packing factor  $D$  is about 80. Using a typical CMOS value  $V_{\text{READ}} = 1\text{V}$ ,<sup>5</sup> the smallest possible  $R_{\text{sense}}$  may be found from Eq. (22) and (23) to equal  $3 \text{ K}\Omega$ , i.e. a level much lower than the parallel resistance of the semi-infinite ladder  $(R_{\text{wire}}R_{\text{OFF}}/D)^{1/2} \approx 84 \text{ K}\Omega$ . It is easy to check that all conditions (20)-(22) are satisfied, so that our approximate formulas are indeed valid. The corresponding intrablock delay in this case is about 55 ps, i.e. much smaller than that of the ECC decoding and hence can be neglected. (Even in case of  $F_{\text{CMOS}} = 22$  nm and  $F_{\text{nano}} = 2.2$  nm the intrablock delay, which is linearly proportional to the  $R_{\text{wire}}$ , is only 400 ps.)<sup>78</sup>

Similarly, the interblock delay (25) can be neglected since it is below 1 ns even for the “worst” case of  $F_{\text{CMOS}} = 22$  nm. Indeed, for CMOS metal 1 layer lines, the typical specific resistance is about  $40 \Omega/\mu\text{m}$ ,<sup>5</sup> i.e. for the block size  $W = 256$ , the CMOS wire resistance  $R_{\text{CMOS}}$  is approximately  $700 \Omega$ . Assuming  $F_{\text{nano}} = 2.2$  nm and  $N = 1$  Tbit (which gives  $\sqrt{L}$  of the order of 1000), the corresponding interblock delay is about 0.65 ns, i.e. also much smaller than the ECC decoding delay.

Due to this fact, our results are rather insensitive to the hardware assumptions made above.

## 6.7. Power

The assumptions and formulas given above allow also a calculation of power consumption in all components of the memory. Such calculations show that the power consumption is well

below the ITRS-specified limits.<sup>5</sup> Indeed, the consumption is dominated by the static power dissipated in nanowires connected to the nanodevices in ON state. Since there might be at most  $r^2$  such nanowires (with the average resistance  $1/2r^3R_{\text{wire}}$ ) in a block, the corresponding power for the whole memory can be estimated as  $2\sqrt{L}(V_{\text{READ}})^2/(rR_{\text{wire}})$ , giving the power density well below  $1 \text{ W/cm}^2$  for all cases we have studied. Note, however, that local overheating is still possible and should be carefully evaluated in future. At such analysis, one may consider an option of a modest increase of  $R_{\text{ON}}$  and/or decrease of  $V_{\text{READ}}$ , since the resulting increase of the intrablock latency would not affect the total memory access time significantly.

## 7. OPTIMIZATION

Requiring that the total yield  $Y$  described by Eq. (5) is fixed at a certain level, we first investigate how the normalized memory area per useful bit, defined as<sup>36</sup>

$$a \equiv \frac{A}{N(F_{\text{CMOS}})^2}, \quad (26)$$

depends on the block size  $W$ . Figure 9 shows the results of this calculation for fixed granularity and ECC code parameters. Not surprisingly, the exponential explosion of redundancy at  $W \rightarrow \infty$ , pertinent to the global crossbar memory architecture,<sup>36</sup> disappears in our current approach, since the granularity of exclusion does not depend on the block size.

Concerning other contributions to  $A$ , for large arrays (in the case shown in Fig. 9, for  $W > 100$ ), the most important contributor is the mapping table whose area grows logarithmically with  $W$  – see Eq. (13). This overhead can be reduced dramatically by choosing appropriate granularity parameter  $g$  – see below. The next important overhead is that of the cell address decoders, which does not allow the reduction of  $W$  below  $\sim 2^7$ . Because of that, in our calculations we have used the fixed value  $W = 2^8 = 256$ . This is very convenient for the most interesting case  $F_{\text{CMOS}}/F_{\text{nano}}$

$=10$  ( $r = 16$ ), because in this case  $W = r^2 = 256$ , and there is no need in the barrel shifter (Fig. 4). Moreover, in this case, and  $g \geq 1$ , all CMOS data wires of each array are fully used at each operation, resulting in the maximum possible throughput.

Let us now return to the mapping table overhead. Figure 10 shows this overhead in comparison with that due to the redundant cells, as a function of data fragment size (granularity). As Eq. (13) shows, when the granularity is increased, the mapping table shrinks (because the memory has fewer fragments). On the other hand, the redundancy overhead is skyrocketing, because the use of heavy BCH codes, which are necessary to handle such long fragments, is very expensive in terms of decoding latency. Only if the fraction of bad bits is very low, this increase of the redundant cell number (in order to get fixed yield  $Y$ ) is deferred to very high data fragment size.

## 8. RESULTS AND DISCUSSION

Figure 11 presents typical final results<sup>79</sup> of our optimization procedure carried out for several values of the total access time (for our parameters, dominated by the ECC decoding time). The cusps on the curves are due to sudden changes of discrete parameters ( $gr^2$ ,  $n$  and  $k$ ) for which the largest memory density is achieved. These changes are clearly visible in Table 1 which provides details of the optimization results for the particular case  $\tau = 10$  ns and  $F_{\text{CMOS}}/F_{\text{nano}} = 10$ . One can see that as the fraction  $q$  of bad memory cells is increased, the granularity  $gr^2$  has to be decreased in order to sustain acceptable probability of ECC-correctable data fragments. (Only for very high  $q$  the fragment length  $gr^2$  becomes less than the nanowire segment size  $r^2$ , i.e. the fragment may be stored in a single block.) As a result, optimal error correcting codes become shorter, i.e.  $n$  and  $k$  decrease.

Returning to Fig. 11a, it shows that even at the initial stage of the CMOL technology development (when the ratio  $F_{\text{CMOS}}/F_{\text{nano}}$  is of the order of 3), the defect tolerance and density of crossbar memories may be quite impressive. Indeed, if the required latency is not too small (say, 10 ns or higher), crossbar memories may become denser than CMOS-based memories at the fraction of bad devices as high as  $\sim 15\%$ , and at the (quite realistic) value  $q = 5\%$  provide a nearly five-fold density edge ( $\alpha \cong 1.2$  instead of 6). For  $F_{\text{nano}} = 15$  nm this would mean useful a density of  $\sim 30$  Gbits/cm<sup>2</sup>, i.e. the level which CMOS technology may be able to reach in the very end of scaling,<sup>5</sup> if ever. Note that these (and all the following) results are much better than those obtained in our initial work,<sup>36</sup> where, for example, the normalized area  $\alpha = 1.2$  could only be reached (for the similar  $F_{\text{CMOS}}/F_{\text{nano}}$  ratio) at  $q \cong 3\%$  using the impracticably long Exhaustive Search algorithm, while for the realistic Repair Most algorithm,  $q_{\text{max}}$  was as low as  $\sim 10^{-3}$ . This improvement is due mostly to the use of more advanced error correcting codes.

Moreover, as the CMOL technology matures, and the  $F_{\text{CMOS}}/F_{\text{nano}}$  ratio approaches an order of magnitude (say,  $F_{\text{CMOS}} = 32$  nm,  $F_{\text{nano}} = 3$  nm<sup>80</sup>), the crossbar memory superiority may be quite spectacular. Indeed, as Fig. 11b shows, for the defect fraction  $q = 2\%$  (which looks quite plausible), the cell area factor  $\alpha$  may become as low as 0.1, implying the dimensional density as high as 1 Tbit/cm<sup>2</sup>, far beyond the most optimistic projections for CMOS technology.<sup>5</sup>

For the latter design point, the estimated memory throughput is equally impressive (helped by the fact that this pitch ratio optimizes the throughput, since all CMOS data lines are being utilized). Indeed, even assuming a BCH decoder without pipelining, the memory bandwidth can be as high as 30 Tb/s for a 1 cm<sup>2</sup> chip. (For reaching this maximum, the BCH decoder has to be replicated for each column of CMOL blocks, but even in this case the corresponding area overhead is less than 20% - Fig. 7).

Note, however, that our optimistic results for the memory speed (latency and throughput) are based on the fundamental physical limitations for the crosspoint nanodevice parameters, in particular,  $R_{ON}$ . For the currently implemented programmable diodes, the picture is somewhat different. For example, for the simple and reproducible  $\text{CuO}_x$  devices,<sup>15</sup> scaled down to  $F_{\text{nano}} = 3$  nm, the effective value of  $R_{ON}/D$  would be  $\sim 2 \text{ M}\Omega$ , resulting in intrablock latency of about 50 ns. This means that our results (Fig. 11) would degrade only slightly. On the other hand, for the demonstrated reproducible molecular monolayers,<sup>58</sup> typical  $R_{ON}/D$  of a similarly scaled crosspoint device would be in the  $\text{G}\Omega$  range, so that the memory speed would be much lower. Nevertheless, a considerable progress of the improvement of molecular programmable diodes during the next few years may be readily anticipated.

Another reservation has been made concerning our defect-tolerance results. Indeed, our analysis has been limited to the crosspoint device defects equivalent to the stuck-at-open faults, while in practice defects of other types are also possible. In this context, it is worth mentioning that both breaks of the lower-layer nanowires and defective pins leading to these nanowires can be very efficiently excluded by reconfiguration around them, using the techniques described above. The exclusion of defects in the upper layer nanowires requires some modifications of the hardware. For example, by increasing the size of array it is possible to access nanodevices through several pins contacting the same nanowire simultaneously. (This would require a somewhat more complicated address control circuitry, but the area of these circuits would still give a negligible contribution to the total area.)

In contrast, the worst-case stuck-on-close-type defects (resulting in a very low crosspoint resistance) can be overcome by exclusion of both nanowires leading to the defective crosspoint. Preliminary estimates show that such exclusion incurs high area overhead, so that our current

architecture is not very efficient for dealing with these defects. On the other hand, if a stuck-on-close defect has a resistance comparable with the nominal value of  $R_{ON}$ , addressing of several adjacent nanodevices may still be possible. (An analysis of this problem is similar to finding acceptable margins for variations of  $R_{OFF}$ ,  $R_{ON}$ , and  $R_{wire}$ .) A quantitative analysis of the memory tolerance to such defects is one of our next goals.

Finally, let us repeat that though our calculations have been carried out for CMOL memories with their segmented nanowire structure (Fig. 4), we have found the contribution from the nanowire recharging time to the total access time negligible.<sup>81</sup> This is why our final results are actually valid also for crossbar memories with global blocks and peripheral nano/CMOS interfaces, though they seem much harder for the practical implementation than the distributed CMOL interface.

#### ACKNOWLEDGMENTS

Useful discussions of various aspects of digital CMOL circuits with J. Barhen, S. Das, A. DeHon, S. Dirk, J. Ellenbogen, P. Franzon, S. Goldstein, D. Hammerstrom, R. Karri, R. Kiehl, P. Kuekes, J. Lukens, A. Mayr, A. Orailoglu, G. S. Snider, M. Stan, D. Tennant, R. van Zee, K. Wang, R. S. Williams, T. Zhang, and N. B. Zhitenev are gratefully acknowledged. The work has been supported in part by AFOSR, DTO, and NSF.

## APPENDIX: BCH DECODER

### A. General structure

For simplicity, let us consider primitive BCH codes, with the code length on the finite field  $\text{GF}(2^m)$  is  $k = 2^m - 1$ . A plausible implementation of the fast decoding for binary BCH codes requires three major steps:<sup>64,82</sup>

Step 1: syndrome evaluation;

Step 2: finding the error-location polynomial using the Berlekamp-Massey iterative algorithm; and

Step 3: finding error-location numbers with subsequent error-correction.

Let us estimate the area and delay contributions of the circuits of each step.

### B. Step 1: Syndrome calculation

In a completely bit-parallel implementation each bit of a syndrome  $\mathbf{S}$  can be obtained by a separate XOR tree with inputs taken from the received code vector:<sup>82</sup>

$$\mathbf{S} = (S_1, S_2, \dots, S_{2t}) = \mathbf{r}\mathbf{H}^T = \mathbf{r} \begin{bmatrix} 1 & 1 & \dots & 1 \\ (\varphi) & (\varphi^2) & \dots & (\varphi^{2t}) \\ \vdots & \vdots & \ddots & \vdots \\ (\varphi)^{k-1} & (\varphi^2)^{k-1} & \dots & (\varphi^{2t})^{k-1} \end{bmatrix}. \quad (27)$$

Here  $\mathbf{r}$  is a received code word of length  $k$ ,  $\mathbf{H}$  is a parity matrix, and  $\varphi$  is a primitive element in  $\text{GF}(2^m)$ , i.e. each column in the matrix in the Eq. 27 can be rewritten as an  $m$  binary columns. On the average, binary columns of  $\mathbf{H}$  are half-filled with ones while in the worst case there is a column with almost all ones. Hence the syndrome calculation circuit will be comprised of total  $2tm$  XOR trees with the average depth  $\log_2(k/2)$  and the worst depth of  $\log_2 k$  (Fig. 12).

$$\tau_1 = \tau_{\text{add}}(k) = \lceil \log_2(k) \rceil \times \tau_{\text{XOR}}, \quad (28)$$

$$A_1 = 2tm \times A_{\text{add}}(k/2) = tkm \times A_{\text{XOR}}. \quad (29)$$

Here by  $\tau_{\text{add}}(k)$  and  $A_{\text{add}}(k)$  we denote, correspondingly, the delay and area of 1-bit parallel addition in finite field (XOR tree) of  $k$  elements. Similarly,  $\tau_{\text{XOR}}$  and  $A_{\text{XOR}}$  (and also  $\tau_{\text{OR}}$  and  $A_{\text{OR}}$  which are used later in text) denote the delay and area of Boolean logic 2-input XOR (OR) gate, respectively.

### C. Step 2: Finding Error-Location Polynomial with Berlekamp-Massey Algorithm

Since the algorithm is based on a recursive procedure, it cannot be parallelized completely. However, it is possible to speed up the computation of each iteration. The most time-consuming operations in this step are calculating discrepancy  $d$  (element of the field  $\text{GF}(2^m)$ ) and adjusting error location polynomial  $\sigma(X)$  if necessary,<sup>82</sup> e.g., for the  $\mu+1$  step ( $0 \leq \mu \leq t-1$ ):

$$\sigma^{(\mu+1)}(X) = \sigma^{(\mu)}(X) + d_{\mu} d_i^{-1} X^{2(\mu-i)} \sigma^{(i)}(X), \quad (30)$$

$$d_{\mu+1} = S_{2\mu+3} + \sigma_1^{(\mu+1)} S_{2\mu+2} + \sigma_2^{(\mu+1)} S_{2\mu+1} + \dots + \sigma_{j_{\mu+1}}^{(\mu+1)} S_{2\mu+3-j_{\mu+1}}. \quad (31)$$

Here  $i \leq \mu$  and  $j \leq \mu$  are the indices specific to the implementation of the algorithm.<sup>82</sup> For the worst case, one needs to update  $\sigma_{\mu}(X)$  in each iteration  $\mu$ , with the degree of  $\sigma_{\mu}(X)$  exactly equal to  $\mu$ . Hence the calculation of  $\sigma_{\mu}(X)$  in early iterations can be done with smaller circuit area and (slightly) faster calculation time. However it also implies using different hardware for each iteration. Instead we will assume the scheme shown in Fig. 13 where the hardware is shared among the iterations and the latency is the same in all iterations, being dictated by the operations on  $\sigma_{\mu}(X)$  with the maximum degree  $t$ .

The implementation of Eq. 30 (see part 2A of Figure 13) involves one inversion in  $\text{GF}(2^m)$ ,  $2t$  multiplications in  $\text{GF}(2^m)$ ,  $m$  additions of  $t$  bits and one multiplication by the indeterminate  $X^{2(\mu-i)}$  (i.e. a shift operation). Fast inversion can be done directly as a hard-wired LUT table for small  $m$ , so that the complexity of such operation is roughly.<sup>83</sup>

$$\tau_{\text{inv}}(m) = \lceil \log_2(m-1) \rceil \times \tau_{\text{AND}} + (m-1) \times \tau_{\text{OR}}, \quad (32)$$

$$A_{\text{inv}}(m) = \frac{1}{4} m^2 k \times A_{\text{AND}} + \frac{1}{4} m k \times A_{\text{OR}}. \quad (33)$$

For small values of  $m$  (less than 10), the Mastrovito multiplier<sup>84</sup> ensures fast efficient implementation comparable with the other approaches.<sup>83</sup> The complexity of such multiplier is

$$\tau_{\text{mult}}(m) = \tau_{\text{AND}} + 2 \lceil \log_2 m \rceil \times \tau_{\text{XOR}}, \quad (34)$$

$$A_{\text{mult}}(m) = m^2 \times A_{\text{XOR}} + m^2 \times A_{\text{AND}}. \quad (35)$$

Complexity for the fast shift operation (Fig. 13) is

$$\tau_{\text{shift}} = \tau_{\text{AND}} + \lceil \log_2 m \rceil \times \tau_{\text{OR}}, \quad (36)$$

$$A_{\text{shift}} = m t^2 \times A_{\text{AND}} + m t^2 \times A_{\text{OR}}. \quad (37)$$

Thus, the complexity of the part 2A during one iteration is

$$\begin{aligned} \tau_{2A} &= \tau_{\text{inv}}(m) + \tau_{\text{mult}}(m) + \tau_{\text{add}}(t) = \\ &= \left(1 + \lceil \log_2(m-1) \rceil\right) \times \tau_{\text{AND}} + \left(2 \lceil \log_2 m \rceil + \lceil \log_2 t \rceil\right) \times \tau_{\text{XOR}} + (m-1) \times \tau_{\text{OR}}, \end{aligned} \quad (38)$$

$$\begin{aligned} A_{2A} &= A_{\text{inv}}(m) + 2t \times A_{\text{mult}}(m) + A_{\text{shift}} + m \times A_{\text{add}}(t) = \\ &= \left(2m t^2 + 2m^2 t + \frac{1}{4} m^2 k\right) \times A_{\text{AND}} + (m^2 t + m t) \times A_{\text{XOR}} + \left(m^2 t + \frac{1}{4} m n\right) \times A_{\text{OR}} \end{aligned} \quad (39)$$

Here it is assumed that the inversion is a slower operation than the first multiplication and shift, and hence it is included in the critical path delay calculation. Finally, assuming that part 2B is implemented with  $(t - 1)$  multiplications and additions, and neglecting other circuitry (e.g., a control unit), the total complexity of Step 2 is

$$\begin{aligned} \tau_2 &= t \times [\tau_{2A} + \tau_{\text{mult}}(m) + \tau_{\text{add}}(t)] = \\ &= t \left(2 + \lceil \log_2(m-1) \rceil\right) \times \tau_{\text{AND}} + t \left(2 \lceil \log_2 t \rceil + 4 \lceil \log_2 m \rceil\right) \times \tau_{\text{XOR}} + t(m-1) \times \tau_{\text{OR}}, \end{aligned} \quad (40)$$

$$A_2 = A_{2A} + (t-1) \times A_{\text{mult}}(m) + m \times A_{\text{add}}(t) = \left( mt^2 + 3m^2t + \frac{1}{4}m^2k - m^2 \right) \times A_{\text{AND}} + (3m^2t + 2mt - m^2) \times A_{\text{XOR}} + \left( \frac{1}{4}mk + mt^2 \right) \times A_{\text{OR}}. \quad (41)$$

#### D. Step 3: Finding Error Location Numbers and Correction

The simple substitution which is performed in parallel for all  $2^m$  elements is the fastest (though rather area-expensive) way to find the error location numbers. The test circuit for checking whether some element from  $\text{GF}(2^m)$  is a root of the error location polynomial  $\sigma(X)$  requires a multiplication by a constant (Fig. 15) which has a complexity:<sup>83,84</sup>

$$\tau_{\text{const}}(m) = \lceil \log_2 m \rceil \times \tau_{\text{XOR}}, \quad (42)$$

$$A_{\text{const}}(m) = \left( \frac{1}{2}m^2 - m \right) \times A_{\text{XOR}}. \quad (43)$$

Hence the test circuit can be implemented with

$$\tau_{\text{test}} = \tau_{\text{const}}(m) + \tau_{\text{add}}(t) + \log_2(m) \times \tau_{\text{AND}} = \log_2(m) \times \tau_{\text{AND}} + (\lceil \log_2 m \rceil + \lceil \log_2(t) \rceil) \times \tau_{\text{XOR}}, \quad (44)$$

$$A_{\text{test}} = t \times A_{\text{const}}(m) + m \times A_{\text{add}}(t) + m \times A_{\text{AND}} = \frac{1}{2}tm^2 \times A_{\text{XOR}} + m \times A_{\text{AND}}, \quad (45)$$

so that the total complexity of the step 3 is

$$\tau_3 = \tau_{\text{test}} + \tau_{\text{XOR}} = \lceil \log_2(m) \rceil \times \tau_{\text{AND}} + (1 + \lceil \log_2 m \rceil + \lceil \log_2(t) \rceil) \times \tau_{\text{XOR}}, \quad (46)$$

$$A_3 = k \times A_{\text{test}}(m) + k \times A_{\text{XOR}} = \left( \frac{1}{2}tm^2k + k \right) \times A_{\text{XOR}} + km \times A_{\text{AND}}. \quad (47)$$

#### F. Summary

To summarize our results, it is more convenient to express the decoder complexity in technology-independent units such as CMOS half-pitch  $F_{\text{CMOS}}$  and fanout-of-4 inverter delay  $\tau_{\text{FO4}}$ . To simplify the analysis let us assume the static CMOS gate implementation. Using SCMOS rules,<sup>3</sup> the areas of the static minimum-size 2-input OR (6-transistor), 2-input AND (6-

transistor) and 2-input XOR (10-transistor) gates are roughly equal to, correspondingly, 150, 150 and  $250 (F_{\text{CMOS}})^2$ , while their delays (assuming that each gate drives only one copy of itself) are about  $2/3$ ,  $2/3$  and  $9/5$ .<sup>85</sup> Finally, adding contributions from each step, i.e. using Eqs. 28, 29, 40, 41, 46, 47 and dropping negligibly small terms, the full area and latency of the BCH decoder can be described by Eqs. 14 and 15, correspondingly.

It is worth mentioning that our model, though sufficient for our purposes, is rather crude because it: (i) does not account for wire delays; (ii) is based on the minimum-width gates, and (iii) does not use advanced circuitry (faster circuit families). The first issue is probably the most important, and our estimates are probably on the optimistic side of the real tradeoff picture. Nevertheless, they seem sufficient for our purposes.

## REFERENCES

1. J. L. Hennessy and D. A. Patterson, *Computer Architecture: A Quantitative Approach*, Morgan Kaufmann, San Francisco, CA (2003)
2. B. Prince, *Semiconductor Memories*, 2<sup>nd</sup> ed., Wiley, Chichester, UK (1991)
3. J. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits*, 2<sup>nd</sup> ed., Prentice Hall, Upper Saddle River, NJ (2002)
4. W. Wulf and S. McKee, *Comp. Architecture News* 23, 20 (1995)
5. International Technology Roadmap for Semiconductors, 2005 Edition, available online at <http://www.itrs.net/Common/2005ITRS/Home2005.htm>
6. C. J. Amsinck, N. H. Di Spigna, D. P. Nackashi, and P. D. Franzon, *Nanotechnology* 16, 2251 (2005)
7. Actually, only one of the “write 0” and “write 1 “operations can be performed simultaneously with all cells. Because of the opposite polarity of the necessary voltages across nanodevices for these two operations, the complete write may be implemented in two steps, e. g., first writing 0s and then writing 1s.
8. For the review of early results, see, e.g., G. Dearnaley, A. M. Stoneham, and D. V. Morgan, *Rep. Prog. Phys.* 33, 1129 (1970). References 9-15 below are to the most recent work.
9. S. Seo, M. J. Lee, D. H. Seo, E. J. Jeoung, D.-S. Suh, Y. S. Young, I. K. Yoo, I. R. Hwang, S. H. Kim, I. S. Byun, J.-S. Kim, J. S. Choi, and B. H. Park, *Appl. Phys. Lett.* 85, 5655 (2004)
10. I. G. Baek, D. C. Kim, M. J. Lee, H.-J. Kim, E. K. Yim, M. S. Lee, J. E. Lee, S. E. Ahn, S. Seo, J. H. Lee, J. C. Park, Y. K. Cha, S. O. Park, H. S. Kim, I. K. Yoo, U-In Chung, J. T. Moon and B. I. Ryu, *IEDM’05 Tech. Digest*, 31.4 (2005)

11. D. C. Kim, S. Seo, S. E. Ahn, D.-S. Suh, M. J. Lee, B.-H. Park, I. K. Yoo, I. G. Baek, H.-J. Kim, E. K. Yim, J. E. Lee, S. O. Park, H. S. Kim, U-I. Chung, J. T. Moon, and B. I. Ryu, *Appl. Phys. Lett.* 88, 202102 (2006)
12. B. J. Choi, D. S. Jeong, S. K. Kim, C. Rohde, S. Choi, J. H. Oh, H. J. Kim, C. S. Hwang, K. Szot, R. Wasser, B. Reichenberg, and S. Tiedke, *J. Appl. Phys.* 98, 033715 (2005)
13. D. Lee, H. Choi, H. Sim, D. Choi, H. Hwang, M.-J. Lee, S.-A. Seo, and I. K. Yoo, *IEEE Electron Device Lett.* 26, 719 (2005)
14. H. Sim, D. Choi, D. Lee, M. Hasan, C. B. Samantray, and H. Hwang, *Microelectron. Eng.* 80, 260 (2005)
15. A. Chen, S. Haddad, Y.-C. Wu, T.-N. Fang, Z. Lan, S. Avanzino, S. Pangrle, M. Buynoski, M. Rathor, W. Cai, N. Tripsas, C. Bill, M. VanBuskirk, and M. Taguchi, *IEDM'05 Tech. Digest*, 31.3 (2005)
16. L. P. Ma, J. Liu, and Y. Yang, *Appl. Phys. Lett.* 80, 2997 (2002);
17. L. Ma, S. Pyo, J. Ouyang, Q. Xu, and Y. Yang, *Appl. Phys. Lett.* 82, 1419 (2003);
18. J. Ouyang, C. W. C. W. Chu, C. R. Szmanda, L. Ma, and Y. Yang, *Nature Materials* 3, 918 (2004);
19. L. P. Ma, Q. Xu, and Y. Yang, *Appl. Phys. Lett.* 84, 4908 (2004)
20. L. D. Bozano, B. W. Kean, V. R. Deline, J. R. Salem, and J. C. Scott, *Appl. Phys. Lett.* 84, 607 (2004)
21. R. Sezi, A. Walter, R. Engl, A. Maltenberger, J. Schumann, M. Kund, and C. Dehm, *IEDM'03 Tech. Digest*, 10.2.1 (2003)
22. Y.-S. Lai, C.-H. Tu, D.-L. Kwong, and J. S. Chen, *Appl. Phys. Lett.* 87, 122101 (2005)

23. C. P. Collier, E. W. Wong, M. Belohradsky, F. M. Raymo, J. F. Stoddart, P. J. Kuekes, R. S. Williams, and J. R. Heath, *Science* 285, 391 (1999)
24. C. P. Collier, G. Mattersteig, E. W. Wong, Y. Kuo, K. Beverly, J. Sampaio, F. M. Raymo, J. F. Stoddart, and J. R. Heath, *Science* 289, 1172 (2000)
25. Y. Chen, G.-Y. Jung, D. A. A. Ohlberg, X. Li, D. R. Stewart, J. O. Jeppesen, K. A. Nielsen, J. F. Stoddart, and R. S. Williams, *Nanotechnology* 14, 462 (2003)
26. C. Li, D. Zhang, X. Li, S. Han, T. Tang, C. Zhou, W. Fan, J. Koehne, J. Han, M. Meyyappan, A. M. Rawlett, D. W. Price, and J. M. Tour, *Appl. Phys. Lett.* 82, 645 (2003)
27. W. Wu, G.-Y. Jung, D. L. Olynick, J. Straznicky, Z. Li, X. Li, D. A. A. Ohlberg, Y. Chen, S.-Y. Wang, J. A. Liddle, W. M. Tong, and R. S. Williams, *Appl. Phys. A* 80, 1173 (2005)
28. Y-C. Chen, C. F. Chen, C. T. Chen, J. Y. Yu, S. Wu, S. L. Lung, R. Liu, and C-Y. Lu, *IEDM'03 Tech. Digest*, 37.4.1 (2003)
29. C. M. S. Torres, S. Zankovych, J. Seekamp, A. P. Kam, C. C. Cedeno, T. Hoffmann, J. Ahopelto, F. Reuther, K. Pfeiffer, G. Bleidiessel, G. Gruetzner, M. V. Maximov, and B. Heidari, *Materials Science. & Eng. C* 23, 23 (2003)
30. L. J. Guo, *J. Phys. D* 37, R123 (2004)
31. D. J. Wagner and A. H. Jayatissa, *Proc. SPIE* 6002, 136 (2005)
32. S. R. J. Brueck, in: *International Trends in Applied Optics*, SPIE Press, Bellingham, WA (2002), p. 85
33. H. H. Solak, C. David, J. Gobrecht, V. Golovkina, F. Cerrina, S. O. Kim, and P. F. Nealey, *Microelectron. Eng.* 67, 56 (2003)

34. G.-Y. Jung, E. Johnson-Halperin, W. Wu, Z. Yu, S.-Y. Wang, W. M. Tong, Z. Li, J. E. Green, B. A. Sheriff, A. Boukai, Y. Bunimovich, J. R. Heath, and R. S. Williams, *Nano Letters* 6, 351 (2006)
35. We do not include in our comparison the data storage systems (such as hard disk drives, etc.) which cannot be used for bit-addressable memories because of their very large (millisecond-scale) access time.
36. D. B. Strukov and K. K. Likharev, *Nanotechnology* 16, 137 (2005)
37. A. DeHon, S. C. Goldstein, P. J. Kuekes, and P. Lincoln, *IEEE Trans. on Nanotechnology* 4, 215 (2005)
38. M. R. Stan, P. D. Franson, S. C. Goldstein, J. C. Lach, and M. M. Ziegler, *Proc. IEEE* 91, 1940 (2003)
39. S. Das, G. Rose, M. M. Ziegler, C. A. Picconatto, and J. E. Ellenbogen, in: *Introducing Molecular Electronics*, edited by G. Cuniberti, G. Fagas, and K. Richter, Springer, Berlin (2005), p. 479
40. A. DeHon and K. K. Likharev, in: *Proc. ICCAD'05*, IEEE, Piscataway, NJ (2005), p. 375
41. P. J. Kuekes, G. S. Snider, and R. S. Williams, *Sci. Amer.* 293, 72 (2005)
42. K. K. Likharev and D. B. Strukov, in: *Introducing Molecular Electronics*, edited by G. Cuniberti, G. Fagas, and K. Richter, Springer, Berlin (2005), p. 447
43. K. K. Likharev, in: *Nano and Giga Challenges in Microelectronics*, edited by J. Greer, A. Korkin and J. Labanowski, Elsevier, Amsterdam (2003), p. 27
44. K. K. Likharev, *Interface* 14, 43 (2005)
45. D. B. Strukov and K. K. Likharev, in *Proc. FPGA'06*, ACM, New York (2006), p. 131
46. J. G. Simmons and R. R. Verderber, *Proc. Roy. Soc. A* 301, 77 (1967)

47. N. F. Mott and J. H. Davies, *Electronic Properties of Non-Crystalline Materials*, 2nd edition, Oxford Univ. Press, Oxford (1979)
48. N. F. Mott, *Conduction in Non-Crystalline Materials*, 2nd edition, Clarendon Press, Oxford, (1993)
49. K. K. Likharev, A. Mayr, I. Muckra, and Ö. Türel, *Ann. New York Acad. Sci.* 1006, 146 (2003)
50. S. Fölling, Ö. Türel, and K. K. Likharev, in: *Proc. 2001 IJCNN, Int. Neural Network Soc.*, Mount Royal, NY (2001), p. 216
51. P. D. Dresselhaus, L. Ji, S. Han, J. E. Lukens, and K. K. Likharev, *Phys. Rev. Lett.* 72, 3226 (1994).
52. L. Ji, P. D. Dresselhaus, S.Y. Han, K. Lin, W. Zheng and J. E. Lukens, *J. Vac. Sci. Technol. B* 12, 3619 (1994)
53. H. Park, J. Park, A. K. L. Lim, E. H. Anderson, A. P. Alivisatos, and P. L. McEuen, *Nature* 407, 57 (2000)
54. S. P. Gubin, Y. V. Gulyaev, G. B. Khomutov, V. V. Kislov, V. V. Kolesov, E. S. Soldatov, K. S. Sulaimankulov, and A. S. Trifonov, *Nanotechnology* 13, 185 (2002)
55. N. B. Zhitenev, H. Meng, and Z. Bao, *Phys. Rev. Lett.* 88, 226801 (2002)
56. J. Park, A. N. Pasupathy, J. I. Goldsmith, C. Chang, Y. Yaish, J. R. Petta, M. Rinkoski, J. P. Sethna, H. D. Abruna, P. L. McEuen, and D. C. Ralph, *Nature* 417, 725 (2002)
57. S. Kubatkin, A. Danilov, M. Hjort, J. Cornil, J. L. Bredas, N. Stuhr-Hansen, P. Hedegart, and T. Bjornholm, *Nature* 425, 698 (2003)
58. H. B. Akkerman, P. W. M. Blom, D. M de Leeuw, and B. De Boer, *Nature* 441, 69 (2006)

59. T. Dadosh, Y. Gordin, R. Krahne, I. Khirich, D. Mahalu, V. Frydman, J. Sperling, A. Yacoby, and I. Bar-Joseph, *Nature* 436, 677 (2005)
60. K. Chakraborty and P. Mazumder, *Fault-Tolerance and Reliability Techniques for High-Density Random-Access Memories*, Prentice Hall, Upper Saddle River, NJ (2002)
61. C. Stapper and H. Lee, *IEEE Trans. on Computers* 41, 1078 (1992)
62. Such requirement seems necessary for the introduction of the radically new (“disruptive”) memory technology. Note, however, that in our both (previous and current) works we compare crossbar memories with “perfect” semiconductor memories with cell area  $(2F_{\text{CMOS}})^2$ . So far, only nonvolatile (flash) memories have approached this frontier,<sup>5</sup> so that our comparative evaluation is rather conservative.
63. F. Sun and T. Zhang, *Two Fault Tolerance Design Approaches for Hybrid CMOS/Nanodevice Digital Memories*, (in press) in *Proc. NANOARCH’06* (2006)
64. R. E. Blahut, *Algebraic Codes for Data Transmission*, Cambridge University Press, Cambridge, UK (2003)
65. R. Betz, *Microelectron. Eng.* 80, 249 (2005)
66. As a matter of principle, it is possible to avoid such loss by connecting CMOS lines on both edges of the array in a specific order, i.e., mapping it on a 3D torus. However, for realistic values of  $r$  the number of such connections (which should be made in CMOS) becomes large and makes such method impracticable.
67. Completely random defects are the most difficult to cope with. Any defect clustering enables higher defect tolerance (with the fixed total number of defects) using the synergetic approach in this paper. This is due to the fact that clustered defects can be more effectively dealt with via reconfiguration, i.e. by replacing nanowire segments for the spare ones.

68. G. Snider, P. Kuekes, and R. S. Williams, *Nanotechnology* 15, 881 (2004)
69. A. DeHon and H. Naeimi, *IEEE Design & Test of Comp.* 22, 306 (2005)
70. D. B. Strukov and K. K. Likharev, *Nanotechnology* 16, 888 (2005)
71. M. Masoumi, F. Raissi, M. Ahmadian, and P. Keshavarzi, *Nanotechnology* 17, 89 (2006)
72. M. Alioto and G. Palumbo, *IEEE Trans. on VLSI* 10, 806 (2002)
73. Decoding of BCH codes is based on an iterative recursive procedure which cannot be parallelized efficiently – see Appendix. The decoding of some other codes, e.g., Euclidean geometry or LDPC,<sup>64</sup> which are less efficient in general in the considered range of  $n$ , may be faster for some particular values of  $n$ ,  $k$  and  $t$ . Also, one can take advantage of the 0-1 asymmetry of a particular defect model. Thus, our present results may be certainly improved in future.
74. The actual write delay can be affected by the fundamental relationship between the retention time and write/erase speed in two-terminal devices. For the usual uniform tunnel barriers, the ratio of these two time constants can hardly exceed 9 orders of magnitude, and only for the (so far, hypothetical) structures with optimized barrier profile<sup>75</sup> may be increased to  $\sim 17$  orders of magnitude. Unless such advanced barriers have been implemented, a crossbar memory with a-few-nanosecond write/erase time may need periodic refresh similar to that used in DRAM. Alternatively, when microsecond-scale write time is acceptable, the retention time may be above 10 years, the industrial standard for non-volatile operation.
75. K. K. Likharev, *IEEE Circuits & Devices*, 16, 16 (2000)
76. N. B. Zhitenev, W. Jiang, A. Evbe, Z. Bao, E. Garfunkel, D. M. Tennant, and R. Cirelli, *Nanotechnology* 17, 1272 (2005)

77. G.-L. Ingold and Yu. V. Nazarov, in: Single Charge Tunneling, edited by H. Grabert and M. H. Devoret, Plenum, New York (1992), p. 21
78. Note, however, that an increase in nanowire resistivity by two orders of magnitude would make the intrablock delay of the order of 100 ns, i.e. turn it into the main component of the memory access time. As a result, using for nanowires any material with  $\rho > 10^{-4} \Omega\text{-cm}$  (typical for semiconductor and molecular nanowires) should be avoided.
79. Though formally the results depend on the total memory size  $N$  and yield  $Y$ , they are rather insensitive to these parameters in the range of our interest ( $N \sim 10^{12}$  bits,  $Y \sim 90\%$ ). As Fig. 11 shows, the required memory access time  $\tau$  also has a marginal effect on density, provided that  $\tau$  is not too small.
80. K. K. Likharev, CMOL Technology: Possible Roadmap, document in preparation
81. This would not be true for the hypothetical (and hardly plausible) semiconductor-wire or molecular-wire crossbars in which high nanowire resistance may seriously affect the memory access time.
82. H. Lee, IEEE Trans. VLSI 11, 288 (2003)
83. C. Paar, Efficient VLSI Architectures for Bit-Parallel Computation in Galois Fields, PhD Thesis, Inst. For Experimental Math., U. Essen, Germany (1994)
84. E. D. Mastrovito, VLSI Architectures for Computation in Galois Fields, PhD Thesis, Linkoping University, Dept. of Elec. Eng., U. Linkoping, Sweden (1991)
85. I. Sutherland, R. F. Sproull, and D. Harris, Logical Effort: Designing Fast CMOS Circuits, Morgan Kaufmann, San Francisco, CA (1999)

## FIGURE CAPTIONS

Fig. 1. Crossbar memory: (a) crossbar array structure, (b)  $I$ - $V$  curve of a single nanodevice (schematically) and (c, d) equivalent circuits of the array showing (c) read and (d) write operation for one of the cells (marked A). On panel (c), green arrow shows the useful readout current, while red arrow shows the parasitic current to the wrong output wire, which is prevented by the nonlinearity of the  $I$ - $V$  curve of device A (if the output voltage is not too high,  $V_{\text{out}} < V_t$ ).

Fig. 2. Low-level structure of the generic CMOL circuit: (a) schematic side view (cross-section along the A-A line) and (b) a top view. For clarity, the last panel shows only two adjacent crosspoint devices which may be addressed via pin pairs  $\{1, 2\}$  and  $\{1, 2'\}$ . The figure shows that the CMOS system has a unique access to each nanowire (and hence to each nanodevice) if the nanowire crossbar is rotated relative the interface pin array by a specific angle  $\alpha = \arctan(1/r) = \arcsin(F_{\text{nano}}/\beta F_{\text{CMOS}}) \ll 1$ . Here  $r$  is an integer, and  $\beta$  is the distance between the adjacent pins (leading to the same crossbar level), expressed in terms of the CMOS pitch  $2F_{\text{CMOS}}$ . (For a more detailed discussion of the CMOL topology, see, e.g., Ref. 42.)

Fig. 3. Crossbar memory structure considered in this work: (a) global and (b) block architectures.

Fig. 4. CMOL block of memory cells (for  $r = 4$ ) and addressing of: (a) an interior and (b) a border column of nanowire segments. Each picture shows only one (selected) column of the segments, the crosspoint nanodevices connected to one (selected) segment, and continuous top-level nanowires connected to these nanodevices. (In reality, the nanowires of both layers fill all the array plane, with nanodevices at each crosspoint.) The block arrows indicate the location of CMOS lines activated at addressing the shown nanodevices.

Fig. 5. Possible structure of the CMOS relay cell. Red and blue points indicate the corresponding interface pins (cf. Fig. 2).

Fig. 6. The assumed structure of peripheral circuits: (a) cell decoder, and (b) barrel shift decoder.

Fig. 7. Delay (black lines) and area (blue lines) of a bit-parallel decoder for binary BCH codes as functions of: (a) the number of errors  $t$  the code can correct for several values of code length  $n$ , and (b)  $n$  for several values of  $t$ . The delay is measured in delays of a standard CMOS fan-out-of-four inverter, while the area in the CMOS half-pitch units. For convenience, horizontal lines show the delay of 1 ns and the area of  $1 \text{ mm}^2$ , for the 22 nm and 45 nm ITRS technology nodes (solid and dashed lines, respectively).

Fig. 8. The equivalent circuit for the readout operation.

Fig. 9. The total chip area per one useful memory cell, and its components (all in the units of  $(F_{\text{CMOS}})^2$ ), as functions of CMOL array size  $W$  (at fixed memory size  $N$ ).

Fig. 10. The normalized memory area overheads due to the redundant memory cells and the mapping tables. Each absolute overhead  $A_x$  is normalized as  $A_x/(A_x+A_{\text{useful}})$ , where  $A_{\text{useful}} = 4N(F_{\text{nano}})^2$  is the area of useful memory cells.

Fig. 11. The total chip area per one useful memory cell, as a function of the bad bit fraction  $q$ , for several values of the memory access time  $\tau$  and two typical values of the  $F_{\text{CMOS}}/F_{\text{nano}}$  ratio. (The ratio used for panel (a) is typical for the initial stage of the CMOL technology development, while that on panel (b) should be reached for its later stage.<sup>80</sup>) The horizontal lines indicate the area for “perfect” CMOS and CMOL memories. In the latter case, this line shows our results for negligible  $q$ , while for the former case we use the ITRS data<sup>5</sup> for the densest semiconductor (flash) memories.

Fig. 12. Syndrome calculation (Step 1) circuit block diagram.

Fig. 13. Implementation of Berlekamp-Massey Iterative Algorithm (Step 2).

Fig. 14. Implementation of Step 3: (a) Circuitry for finding error numbers and error correction.

(b) Test circuit block diagram. Note that by using  $k$  test circuits inversion step is not required.

---

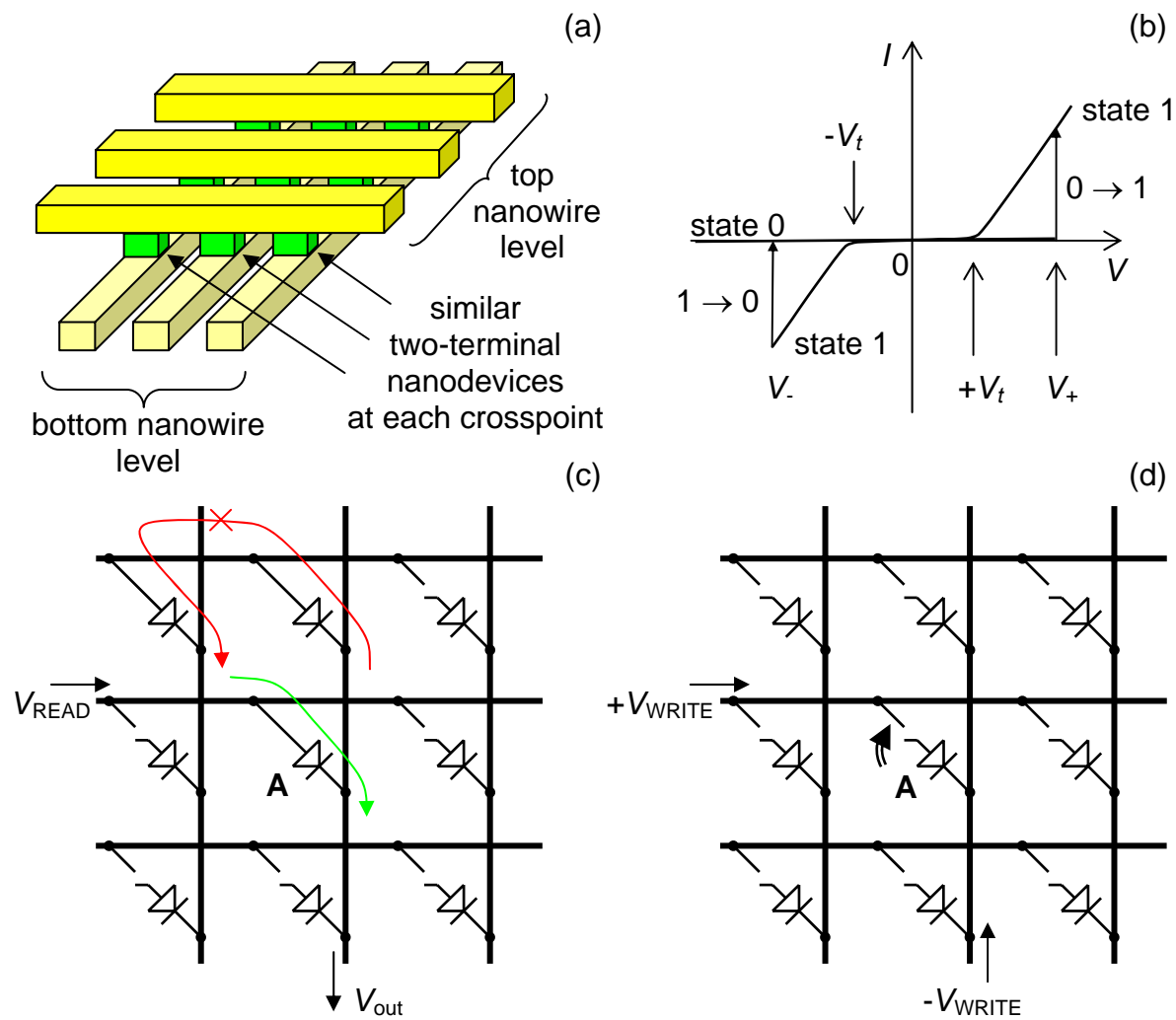


Figure 1

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev

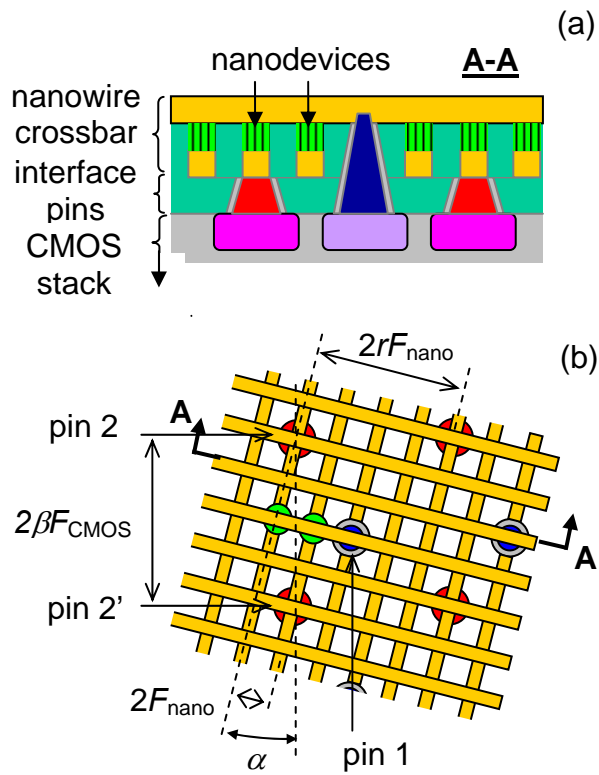


Figure 2

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev



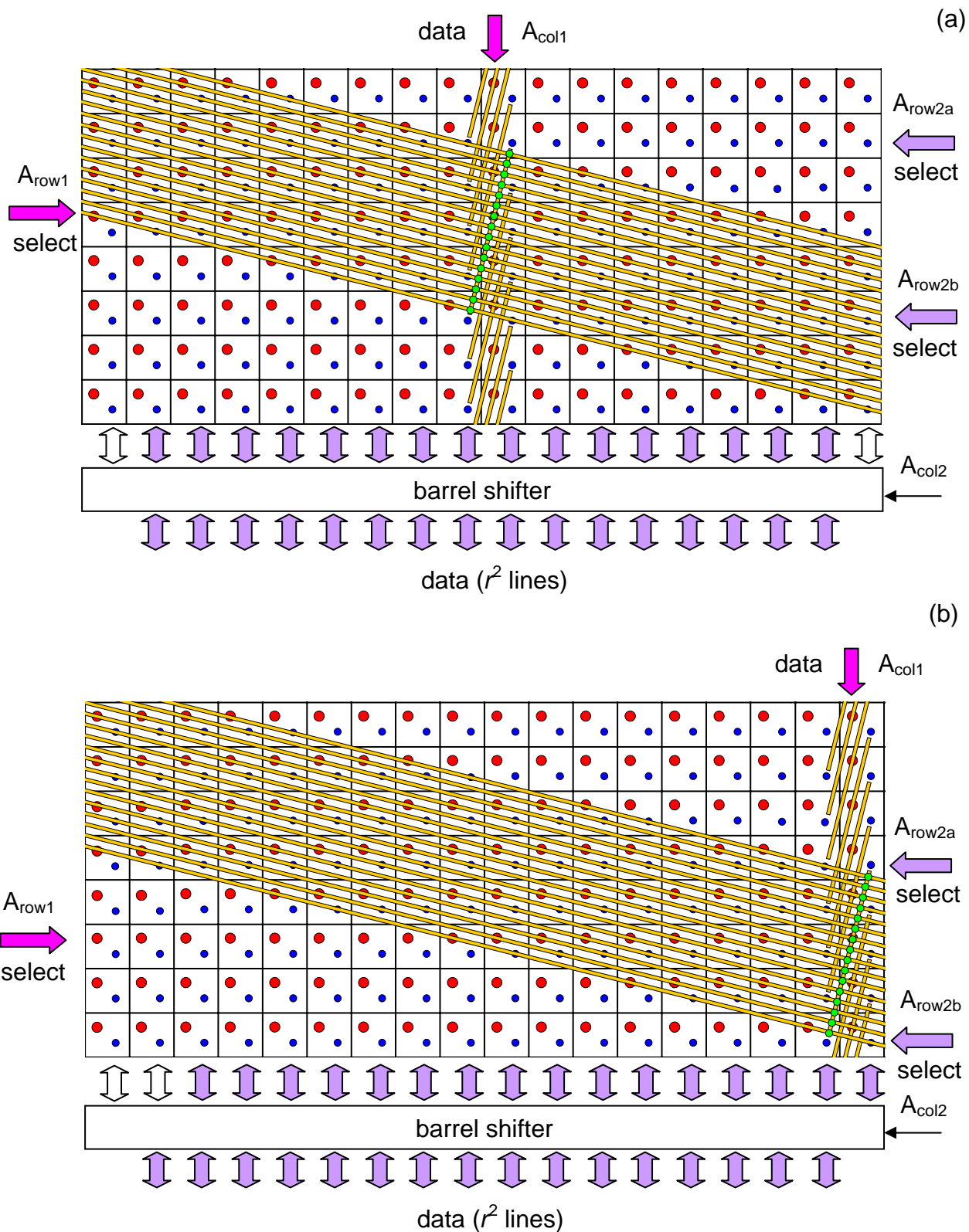


Figure 4

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev

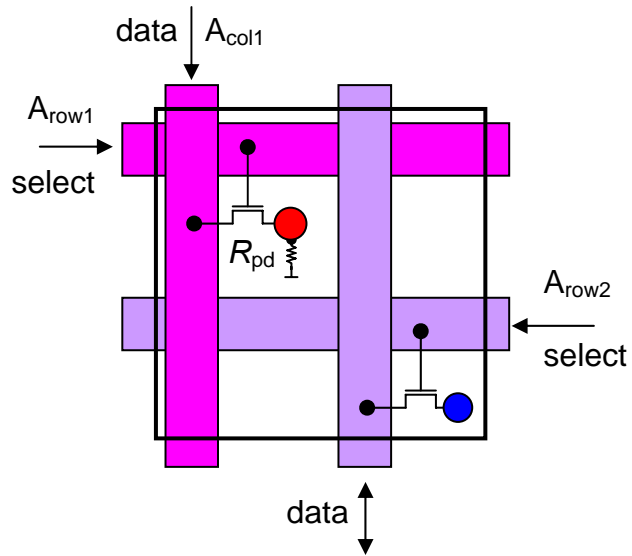


Figure 5

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev

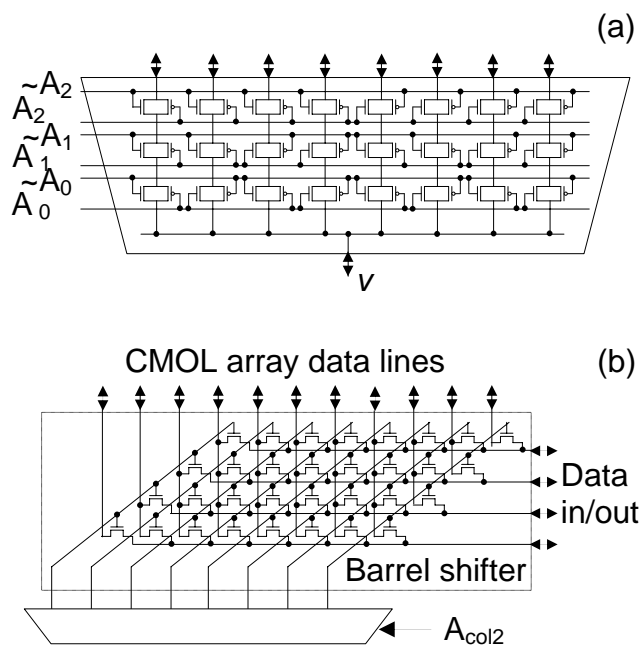


Figure 6

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev

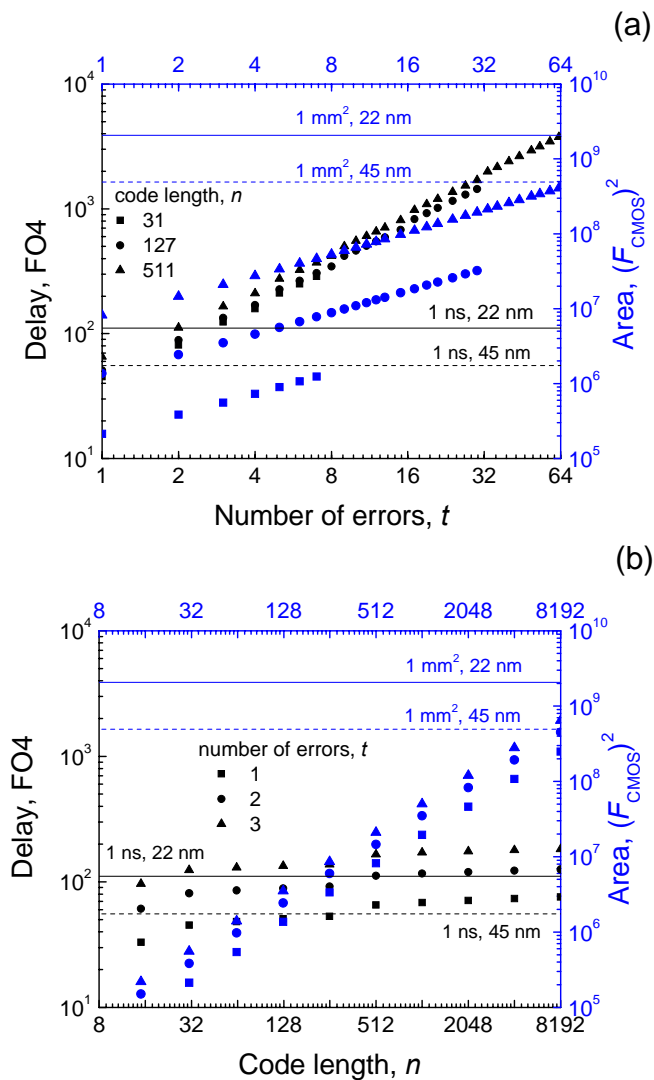


Figure 7

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev

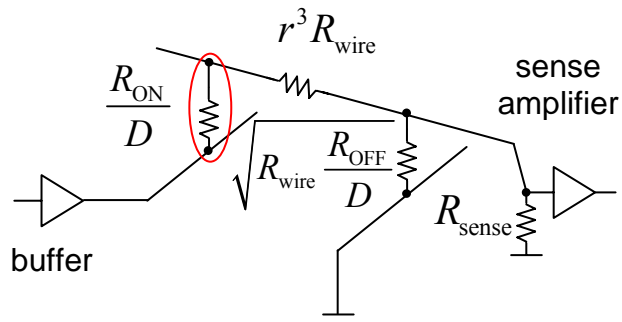


Figure 8

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev

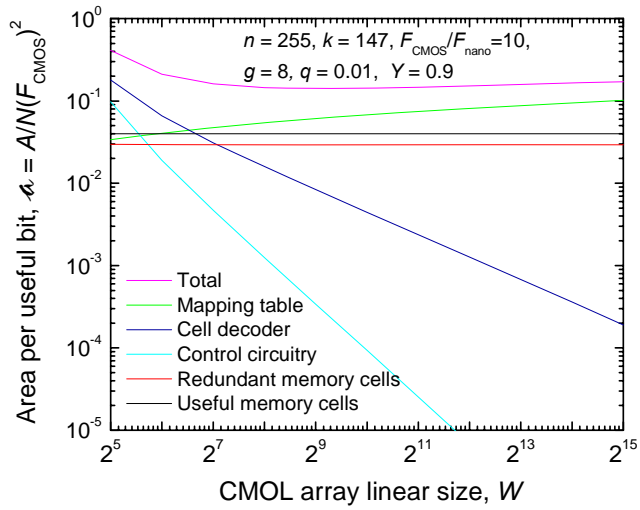


Figure 9

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev

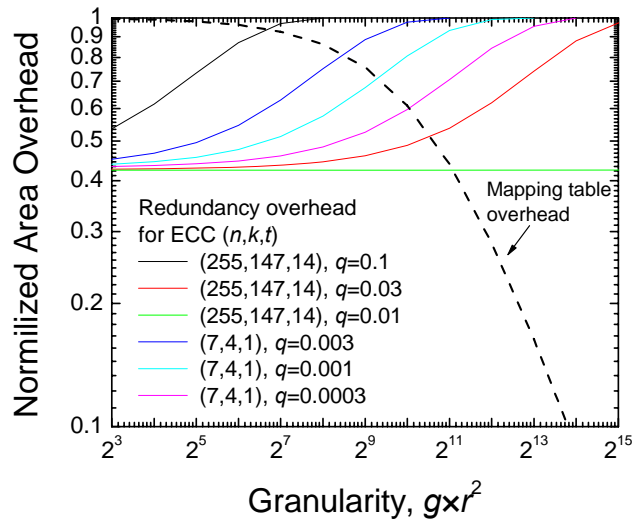


Figure 10

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev

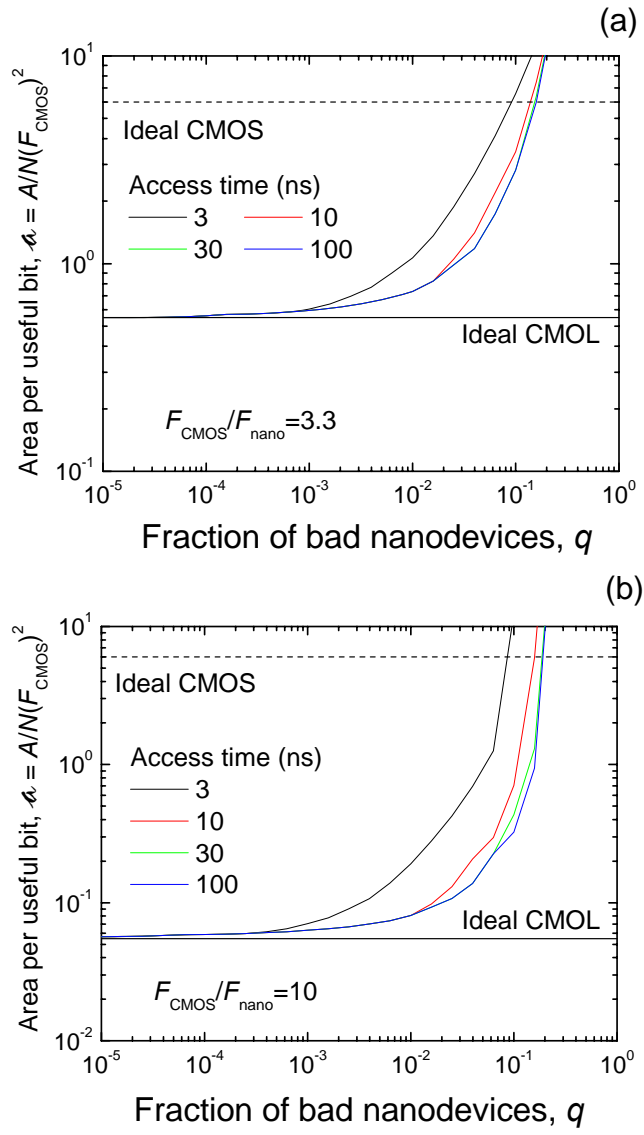


Figure 11

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev

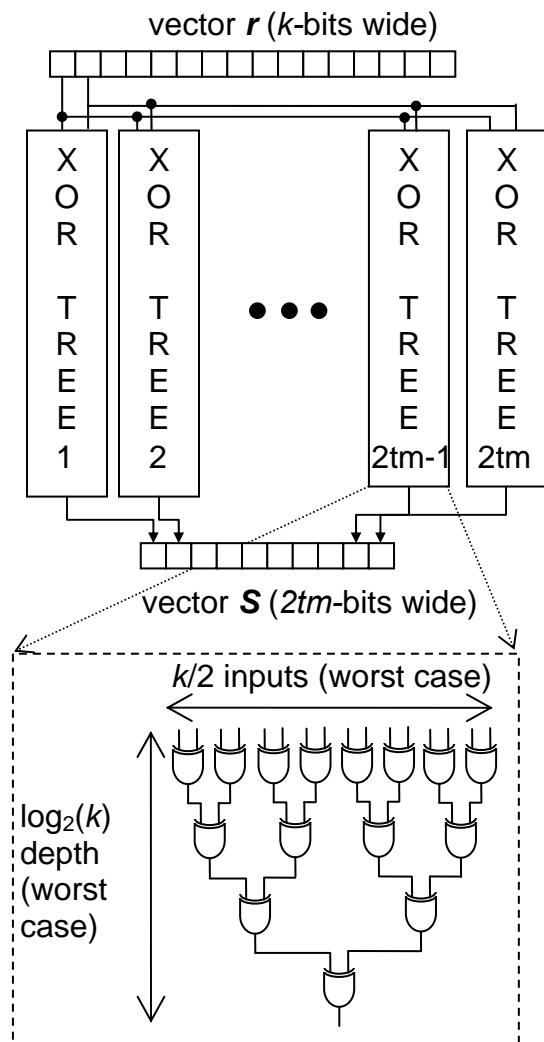


Figure 12

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev

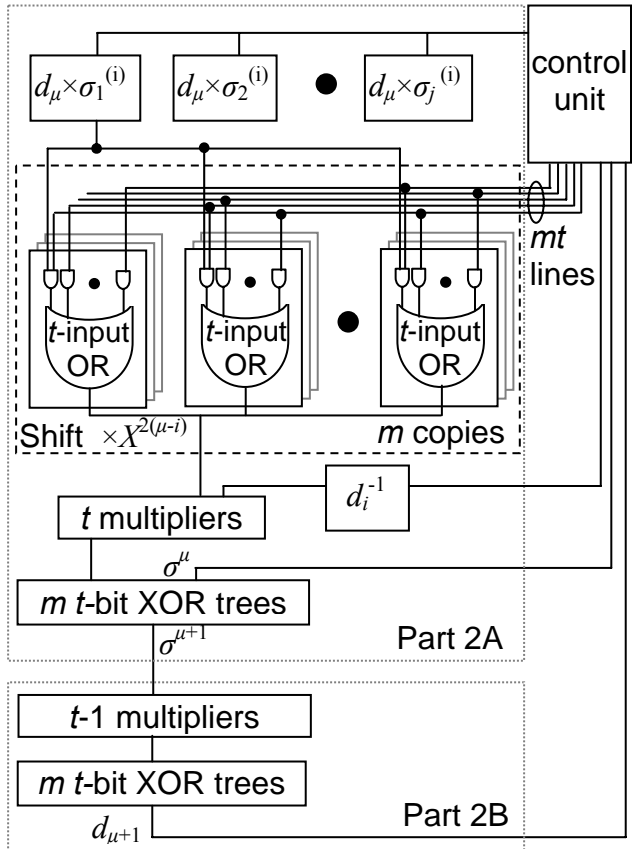


Figure 13

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev

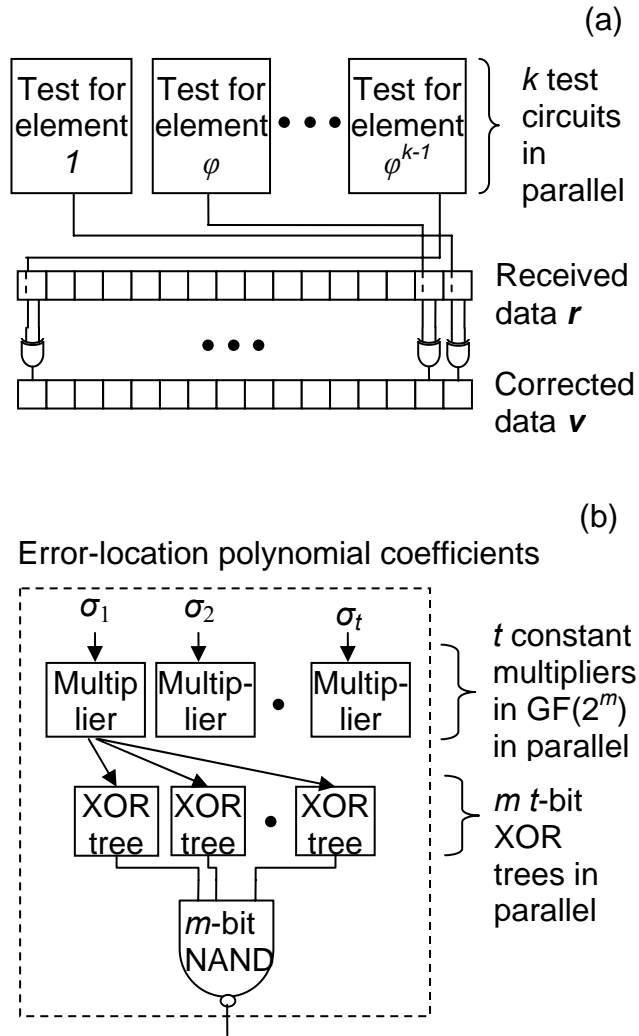


Figure 14

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev

Table 1. Optimal parameters of CMOL memory for the case  $F_{\text{CMOS}} = 45 \text{ nm}$ ,  $F_{\text{nano}} = 4.5 \text{ nm}$ ,  $Y = 90\%$ ,  $N = 1 \text{ Tb}$ , for several values of the fraction  $q$  of bad crosspoint devices

Fraction of bad bits, $q$	Redundant segments per block, $a$	Useful segments per block, $m$	Granularity, $g^2$	ECC useful bits, $k$	ECC total bits, $n$	ECC correctable errors, $t$	ECC latency, $\tau$ (ns)	Area per useful bit, $a$
0.00001	136	61063	131072	239	255	2	1.70	0.057
0.00003	1108	60091	131072	239	255	2	1.70	0.058
0.00010	116	61083	131072	231	255	3	2.55	0.059
0.00032	1389	59810	65536	231	255	3	2.55	0.06
0.00100	271	60928	131072	215	255	5	4.27	0.063
0.00316	724	60475	131072	199	255	7	5.77	0.069
0.01000	2433	58766	32768	179	255	10	8.71	0.081
0.03162	9266	51933	8192	57	127	11	9.36	0.156
0.10000	33698	27501	1024	16	63	11	9.19	0.707

Manuscript title: Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories  
 Manuscript authors: Dmitri B. Strukov and Konstantin K. Likharev