Defect-Tolerant Demultiplexer Circuits Based on Threshold Logic and Coding

Ron M Roth[1] ‡, **Warren Robinett**[2], **Philip J Kuekes**[2] **and R Stanley Williams**[2]

[1] Computer Science Department, Technion, Haifa 32000, Israel.
[2] Hewlett–Packard Laboratories, Palo Alto, CA 94304, USA.

E-mail: ronny@cs.technion.ac.il, warren.robinett@hp.com, philip.kuekes@hp.com and stan.williams@hp.com

Abstract. A defect-tolerant design is presented for a demultiplexer circuit that is based on threshold logic. The design uses coding both to handle (i.e., tolerate) defects in the circuit and to improve the voltage margin in its gates. The following model is assumed for the defects: configured junctions can become either stuck open or stuck closed, and non-configured junctions can become shorted. Two realizations of the circuit are presented: one using conventional transistor circuitry, and the other using nano-scale components and wiring. The design presented in this paper demonstrates how a standard digital building-block circuit—a demultiplexer—can be efficiently protected against several types of defects simultaneously.

Keywords: Constant-weight codes; Defect-tolerance; Demultiplexer; Error-correcting codes; Resistor logic; Threshold logic.

1. Introduction

As circuit components approach the molecular scale—length scales of ∼10nm or less—it becomes increasingly important to design circuits which are *defect-tolerant*, that is, circuits which are able to operate correctly in spite of permanent failures in the circuit components and wiring. A circuit of particular importance for nano-scale computing is the demultiplexer, which can function as an interface subsystem, at the boundary between nano-scale circuitry and conventional micro-scale (e.g. optical-lithography-fabricated CMOS) circuitry. This interface is important because it is likely that both nano-circuitry and CMOS circuitry will be used together in the near-term attempts to capitalize on the density advantages of nano-circuitry. Such a role of demultiplexers is compatible with the configurable crossbar circuits which have been widely used in explorations of nano-circuitry, due to their simple, regular geometry.

Designs for defect-tolerant demultiplexers have been proposed for various platforms, such as diode logic [12], [13], [15], [16], transistor logic [24], and resistor logic [17], [18].

‡ This work was done while visiting Hewlett–Packard Laboratories, Palo Alto, CA 94304, USA.

In these references, defect tolerance is achieved partly by borrowing concepts from the theory of error-correcting codes [20]. Most of the techniques that are described in these references are capable of handling "stuck-open" defects, causing a junction which was configured (with a diode, transistor, gate, or resistor) to become disconnected. The paper [17] analyzes the advantage of using coding in improving the voltage margin of the circuit.

Other types of defects appear to be more challenging to handle. These include "stuck-closed" and "short" defects, causing an unwanted increase in the conductance of a configured junction ("stuck closed") or of a non-configured one ("short"). The recent paper [24] proposes to handle "stuck-closed" defects by replicating in series the transistor or gate at each configured junction. This solution, however, requires to (at least) double the number of gates in the demultiplexer.

In this paper, we present a design for a defect-tolerant demultiplexer based on *threshold logic* (TL) gates [2]. The combination of such gates with coding, in turn, will allow us to handle all previously-mentioned defects: "stuck open", "stuck closed", and "short". Furthermore, we demonstrate how voltage margins can be improved as well.

The paper is organized as follows. In Section 2, we present and analyze an abstract model of our TL-based demultiplexer; our aim in the abstraction is to make the analysis simpler, and more general. Then, in Section 3, we present a realization of the abstract model using transistor logic. Finally, in Section 4, we turn to a second realization, based on resistor logic. As we show, the behavior of the latter realization does deviate in some aspects from the circuits of Sections 2 and 3; still, the underlying design parameters that determine the defect tolerance of all realizations are the same.

Specifically, our realization of a demultiplexer in Section 4 is built using a crossbar, with configurable resistors (but no transistors) at the crosspoint junctions; as such, this realization lends itself to nano-scale implementation and, in particular, can be part of an interface that connects conventional micro-scale circuitry with non-scale circuitry; this interface, in turn, may be prone to defects (see, for example, Figure 1 in [14], and the defects models described in [10], [21], and [22]).

The realization in Section 3, on the other hand, does include transistors as well. While such a realization may be implemented in the future in the nano-scale level, it is currently motivated primarily by traditional (micro-scale) applications. Indeed, some methods for fabricating micro-scale circuitry, such as roll-to-roll imprint fabrication of flexible circuits, are inexpensive, but much less reliable than traditional lithographic manufacturing (for example, LCD displays, which use demultiplexers as row and column decoders, can be fabricated in this way [24, Section IV-C]); by incorporating defect tolerance into the demultiplexer circuits, a significantly higher rate of defects can be allowed in the circuit, thereby resulting in a cheaper manufacturing method.

The exposition in the next section is meant to lay out a unified approach that may suit various applications where demultiplexers are used, both in the micro-scale and nano-scale levels.

2. Abstract model of a TL-based demultiplexer

In this section, we introduce and analyze our abstract model of a demultiplexer that is based on threshold logic (TL) gates (see Figure 1).

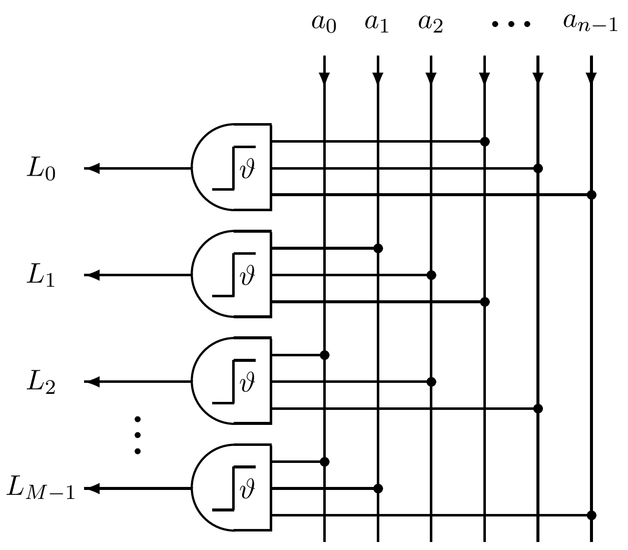


Figure 1. Abstract model of a TL-based demultiplexer.

The input to the demultiplexer consists of n *address lines*, and the output is generated through M *memory lines*§. All lines are binary, taking values on $\{0, 1\}$. We denote the binary word that is formed by the n input bits by $\boldsymbol{a} = a_0 a_1 \ldots a_{n-1}$, and for $i = 0, 1, \ldots, M-1$, we let L_i denote the output at memory line i.

Fixing some positive integer ϑ, we connect the address lines to the memory lines via *ϑ-threshold gates*, which are defined as follows. Each gate can have an arbitrary number of input bits (fan-in) and produces one output bit, which equals 1 if and only if all but at most ϑ input bits are equal to 1 (the threshold value ϑ is assumed to be the same for all gates). Thus, a 0-threshold gate is simply a multiple-input AND gate.

Each memory line is assigned an n-bit *address configuration*, and we denote the address configuration associated with memory line i by

$$\boldsymbol{h}_i = h_{i,0} h_{i,1} h_{i,2} \ldots h_{i,n-1} \ .$$

Let $\mathsf{w}(\boldsymbol{h}_i)$ stand for the Hamming weight of $\boldsymbol{h}_i$ (namely, $\mathsf{w}(\boldsymbol{h}_i)$ is the number of nonzero entries in $\boldsymbol{h}_i$). The value L_i of memory line i is produced as an output of a ϑ-threshold gate whose fan-in equals $\mathsf{w}(\boldsymbol{h}_i)$, and the gate is fed with the values a_j that correspond to address lines j such that $h_{i,j} = 1$.

§ We are using the terms "address line" and "memory line" to be compatible with the widespread application of demultiplexers in memory addressing. However, we do this for convenience only: the design presented here is not limited to any particular application.

Example 2.1. The specific layout shown in Figure 1 corresponds to $n = 6$ address lines and $M = 4$ memory lines, with the following address configurations:

$$\boldsymbol{h}_0 = 000111,\ \boldsymbol{h}_1 = 011100,\ \boldsymbol{h}_2 = 101010,\ \boldsymbol{h}_3 = 110001.$$

In this example, $\mathsf{w}(\boldsymbol{h}_i) = 3$ for all i and, so, each ϑ-gate has fan-in 3. □

Denoting by $\langle \boldsymbol{h}_i, \boldsymbol{a} \rangle$ the inner product of $\boldsymbol{h}_i$ and $\boldsymbol{a}$ (when regarded as real vectors in $\mathbb{R}^n$), we get that

$$L_i = 1 \quad \Longleftrightarrow \quad \langle \boldsymbol{h}_i, \boldsymbol{a} \rangle \geq \mathsf{w}(\boldsymbol{h}_i) - \vartheta\ . \tag{1}$$

But $\mathsf{w}(\boldsymbol{h}_i)$ can be written as $\langle \boldsymbol{h}_i, \mathbf{1} \rangle$, where $\mathbf{1}$ is the all-one vector. Hence, by the linearity of the inner product we get that

$$L_i = 1 \quad \Longleftrightarrow \quad \langle \boldsymbol{h}_i, \mathbf{1}{-}\boldsymbol{a} \rangle \leq \vartheta\ .$$

Since $\mathbf{1}{-}\boldsymbol{a}$ is simply the bitwise complement $\overline{\boldsymbol{a}}$ of $\boldsymbol{a}$, we can rewrite the latter relation as

$$L_i = 1 \quad \Longleftrightarrow \quad \langle \boldsymbol{h}_i, \overline{\boldsymbol{a}} \rangle \leq \vartheta\ . \tag{2}$$

When $L_i = 1$ (respectively, $L_i = 0$) we say that memory line i is *selected* (respectively, *deselected*). A fundamental requirement from any demultiplexer is that—

> For every memory line $\ell \in \{0, 1, \ldots, M{-}1\}$ there is a choice for $\boldsymbol{a}$ such that memory line ℓ is selected and all other lines are deselected.

Since $\langle \boldsymbol{h}_\ell, \overline{\boldsymbol{h}_\ell} \rangle = 0$, we get from (2) that memory line ℓ will be selected if we take $\boldsymbol{a} = \boldsymbol{h}_\ell$. However, we also need to guarantee that all other memory lines are then deselected. We do this through the assignment of the address configurations to the memory lines: obviously, distinct memory lines should be assigned distinct address configurations, yet apparently this requirement is not sufficient for unique selection.

We first consider in Section 2.1 the special case $\vartheta = 0$. Then, in Sections 2.2–2.3, we will make a more general statement that applies to larger ϑ as well. While increasing ϑ imposes more constraints on the set of address configurations (to avoid multiple selection of memory lines), it will also enhance the capability of overcoming defects that the demultiplexer may be subject to. Thus, as a third step, we will present in Section 2.4 a sufficient condition for unique selection also under a certain (well defined) model of defects.

2.1. The zero-threshold case

We introduce the following definition. Given two binary words $\boldsymbol{u} = u_0 u_1 \ldots u_{n-1}$ and $\boldsymbol{v} = v_0 v_1 \ldots v_{n-1}$ in $\{0, 1\}^n$, we say that $\boldsymbol{v}$ *covers* $\boldsymbol{u}$ if for every $0 \leq i < n$,

$$u_i = 1 \quad \Longrightarrow \quad v_i = 1$$

(equivalently, $u_i \leq v_i$); we then write $\boldsymbol{u} \subseteq \boldsymbol{v}$.

When $\vartheta = 0$, memory line ℓ will be selected if and only if the input word $\boldsymbol{a}$ at the address lines covers $\boldsymbol{h}_\ell$. Since all other memory lines must be deselected when line ℓ is selected, the address configurations must satisfy the condition

$$\boldsymbol{h}_i \not\subseteq \boldsymbol{h}_\ell \quad \text{for every } i \neq \ell \tag{3}$$

(in fact, this condition is necessary also when $\vartheta > 0$).

Conversely, when $\vartheta = 0$, condition (3) is also sufficient for allowing the unique selection of memory line ℓ: taking $\boldsymbol{a} = \boldsymbol{h}_\ell$ will then select this line whereas all other memory lines will be deselected.

A nonempty subset $\mathcal{C}$ of $\{0,1\}^n$ is called a *Sperner set* if for every $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{C}$,

$$\boldsymbol{u} \subseteq \boldsymbol{v} \quad \Longrightarrow \quad \boldsymbol{u} = \boldsymbol{v} \ ,$$

namely, no word in $\mathcal{C}$ covers any other word in $\mathcal{C}$. Thus, when $\vartheta = 0$, a necessary and sufficient condition for being able to uniquely select each one of the memory lines is that the set of address configurations forms a Sperner set.

By Sperner's Theorem [27] (see also [19]), the largest Sperner set consists of all words in $\{0,1\}^n$ of Hamming weight $\lfloor n/2 \rfloor$ (or all words of weight $\lceil n/2 \rceil$). The size of the largest Sperner set in $\{0,1\}^n$ is therefore

$$\binom{n}{\lfloor n/2 \rfloor} ,$$

which, for a given number n of address lines, is the largest possible number M of memory lines that allows unique selection for $\vartheta = 0$.

2.2. Unisymmetric distance

The following definitions will be useful when we discuss the criteria for assigning address configurations to memory lines, under the more general setting where the threshold ϑ is not necessarily zero or when the demultiplexer is subject to defects.

Let $\boldsymbol{u}$ and $\boldsymbol{v}$ be two binary words of length n. The *unisymmetric distance* between $\boldsymbol{u}$ and $\boldsymbol{v}$, denoted $\mathsf{d}(\boldsymbol{u}, \boldsymbol{v})$, is defined by

$$\mathsf{d}(\boldsymbol{u}, \boldsymbol{v}) = \min \left\{ \langle \boldsymbol{u}, \overline{\boldsymbol{v}} \rangle, \langle \boldsymbol{v}, \overline{\boldsymbol{u}} \rangle \right\} \ .$$

Note that $\langle \boldsymbol{u}, \overline{\boldsymbol{v}} \rangle$ is the number of positions where $\boldsymbol{u}$ has 1's while $\boldsymbol{v}$ has 0's, and $\langle \boldsymbol{v}, \overline{\boldsymbol{u}} \rangle$ is the respective number when the two words switch roles. The unisymmetric distance is the smallest of these two numbers.

(Referring to the function $\mathsf{d}(\cdot,\cdot)$ as "distance" is actually a misnomer, since this function is not a metric in $\{0,1\}^n$. Indeed, the equality $\mathsf{d}(\boldsymbol{u}, \boldsymbol{v}) = 0$ may hold also when $\boldsymbol{u} \neq \boldsymbol{v}$ (specifically, it holds when either $\boldsymbol{u} \subseteq \boldsymbol{v}$ or $\boldsymbol{v} \subseteq \boldsymbol{u}$); secondly, $\mathsf{d}(\cdot,\cdot)$ does not satisfy the triangle inequality. Nevertheless, for our purposes, our terminology can be justified in that $\mathsf{d}(\cdot,\cdot)$ will measure how "far apart" two binary words should be in order to make them distinguishable even under the presence of defects.)

Given now a subset $\mathcal{C}$ of $\{0,1\}^n$ of size at least 2, we define the *minimum unisymmetric distance* $\mathsf{d}(\mathcal{C})$ of $\mathcal{C}$ as the smallest among the unisymmetric distances between any two distinct words in $\mathcal{C}$:

$$\mathsf{d}(\mathcal{C}) = \min_{\boldsymbol{u},\boldsymbol{v} \in \mathcal{C} :\, \boldsymbol{u} \neq \boldsymbol{v}} \mathsf{d}(\boldsymbol{u},\boldsymbol{v}) \ .$$

A subset $\mathcal{C} \subseteq \{0,1\}^n$ of size M (≥ 2) and minimum unisymmetric distance d will be referred to as an (n, M, d)-*unisymmetric code*, and the elements of $\mathcal{C}$ will be called *codewords*.

Unisymmetric codes were suggested for use in storage applications where the data is subject to error patterns which consist of a combination of symmetric (random) errors and so-called unidirectional errors (hence the adjective "unisymmetric"): see [3], [4], [5], [7], and [23]. These references also contain constructions of unisymmetric codes with guaranteed minimum unidirectional distance, and describe methods for encoding, in a one-to-one manner, unconstrained binary sequences into codewords of the code. In [12], [13], it was shown how such codes can be used in the design of the addressing scheme of diode-based memory systems. For the application of such codes to asynchronous communication, see [8] and [9].

We also recall the notion of a *Hamming distance* between two words $\boldsymbol{u}, \boldsymbol{v} \in \{0,1\}^n$, which is the number of positions where $\boldsymbol{u}$ and $\boldsymbol{v}$ differ. The Hamming distance can be expressed as the *sum*

$$\langle \boldsymbol{u}, \overline{\boldsymbol{v}} \rangle + \langle \boldsymbol{v}, \overline{\boldsymbol{u}} \rangle \ .$$

Now, when $\boldsymbol{u}$ and $\boldsymbol{v}$ have the same Hamming weight, then

$$\langle \boldsymbol{u}, \overline{\boldsymbol{v}} \rangle = \langle \boldsymbol{v}, \overline{\boldsymbol{u}} \rangle = \mathsf{d}(\boldsymbol{u},\boldsymbol{v}) \ ,$$

which means that the Hamming distance between $\boldsymbol{u}$ and $\boldsymbol{v}$ is precisely $2\mathsf{d}(\boldsymbol{u},\boldsymbol{v})$. (This implies that over the set of all words in $\{0,1\}^n$ with a given Hamming weight w, unisymmetric distance *is* in fact a metric.)

An $(n, M, 2d, w)$-*constant-weight code* is a subset $\mathcal{C} \subseteq \{0,1\}^n$ of size M (≥ 2) whose elements all have Hamming weight w, and the minimum Hamming distance between any two distinct codewords is $2d$. It is easy to see that the parameters d and w must satisfy the inequality $d \leq w$. It follows from the previous discussion that every $(n, M, 2d, w)$-constant-weight code is an (n, M, d)-unisymmetric code.

Reference [6] contains constructions of constant-weight codes and bounds on their sizes for a wide range of values n, w, and d (see also [1]). As can be seen in the tables in [6], the largest constant-weight code for given values of n and d is attained therein for $w \in \{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}$ (but we are unaware of a proof that establishes this phenomenon generally).

2.3. General threshold

We now turn to the general case where the threshold ϑ can be any prescribed nonnegative integer. As was the case for $\vartheta = 0$, we will select memory line ℓ by setting the input word $\boldsymbol{a}$ to $\boldsymbol{h}_\ell$. The next result, which follows immediately from (2) and the definition of minimum unisymmetric distance, presents a sufficient condition on the address configurations that guarantees unique selection.

Proposition 2.1. *Let $\mathcal{C} = \{\boldsymbol{h}_0, \boldsymbol{h}_1, \ldots, \boldsymbol{h}_{M-1}\}$ be the set of address configurations assigned to the M address lines of a given demultiplexer as in Figure 1, and suppose that $\mathsf{d}(\mathcal{C})$ is greater than the threshold ϑ. Then for every ℓ in the range $0 \le \ell < M$, setting the input word $\boldsymbol{a}$ to $\boldsymbol{h}_\ell$ will uniquely select memory line ℓ.*

Example 2.2. Referring to the address configurations in Example 2.1, a direct inspection shows that the Hamming distance between any two distinct $\boldsymbol{h}_i$'s is at least 4 (in fact, that distance is exactly 4). Hence, the set of address configurations is a $(6, 4, 4, 3)$-constant-weight code and, as such, it is a $(6, 4, 2)$-unisymmetric code. It follows from Proposition 2.1 that the demultiplexer in Figure 1 (with the parameters of Example 2.1) will uniquely select each memory line even if the threshold value ϑ is taken to be 1. □

2.4. Model of defects

We now consider the tolerance of the demultiplexer in Figure 1 under three types of defects:

Stuck open (`S-open`): an input bit to a ϑ-threshold gate becomes disconnected (from the address line it was connected to) and assumes the value 1.

Stuck closed (`S-closed`): an input bit to a ϑ-threshold gate becomes disconnected and assumes the value 0.

Non-configured short (`N-short`): a new input bit is added to a ϑ-threshold gate and assumes the value 0 (equivalently: the threshold value ϑ of a gate decreases by 1).

(While some of these error models may seem artificial at this point, they will become more concrete when we describe in Section 3 a possible realization of the abstract demultiplexer.)

We have the following result, which exhibits the advantage in choosing the address configurations so that they form a "good" unisymmetric code.

Theorem 2.2. *Let $\mathcal{C} = \{\boldsymbol{h}_0, \boldsymbol{h}_1, \ldots, \boldsymbol{h}_{M-1}\}$ be the set of address configurations assigned to the M address lines of a given demultiplexer as in Figure 1, and let ϑ be the threshold of the gates at the memory lines. For every i in the range $0 \le i < M$, let r_i, s_i, and t_i denote the number of defects of type* `S-open`, `S-closed`, *and* `N-short`, *respectively, at the threshold gate at memory line i, and suppose that for every i, the following two conditions hold:*

(i) $s_i + t_i \le \vartheta$.

(ii) $r_i - t_i < \mathsf{d}(\mathcal{C}) - \vartheta$.

Then for every ℓ in the range $0 \le \ell < M$, setting the input word $\boldsymbol{a}$ to $\boldsymbol{h}_\ell$ will uniquely select memory line ℓ.

Proof. Suppose that memory line ℓ is to be selected, in which case $\boldsymbol{a}$ is set to $\boldsymbol{h}_\ell$. Next, we express the effect of the defects on the entries of $\boldsymbol{a}$, as seen at the inputs of the threshold gate at any given memory line.

For each i, the effect of each defect of type `S-open` at memory line i is to force one entry in $\boldsymbol{a}$ into 1 (even though that entry may originally be 0). Similarly, the effect of each defect of type `S-closed` at memory line i is to force one entry in $\boldsymbol{a}$ into 0. Thus, looking at the input bits of the threshold gate at memory line i, the word $\boldsymbol{a}$ is seen as if it were

$$\boldsymbol{a}_i = \boldsymbol{h}_\ell + \boldsymbol{e}_i - \boldsymbol{e}'_i \, , \tag{4}$$

where $\boldsymbol{e}_i$ (respectively, $\boldsymbol{e}'_i$) is a word in $\{0,1\}^n$ which contains 1 precisely where the defects of type `S-open` (respectively, `S-closed`) make the entries of $\boldsymbol{a}$ $(= \boldsymbol{h}_\ell)$ be seen flipped at memory line i. (The arithmetic in (4) is in $\mathbb{R}^n$.)

Now,

$$\begin{aligned}\langle \boldsymbol{h}_i, \overline{\boldsymbol{a}}_i \rangle &= \langle \boldsymbol{h}_i, \overline{\boldsymbol{h}_\ell + \boldsymbol{e}_i - \boldsymbol{e}'_i} \rangle \\ &= \langle \boldsymbol{h}_i, \overline{\boldsymbol{h}}_\ell - \boldsymbol{e}_i + \boldsymbol{e}'_i \rangle \\ &= \langle \boldsymbol{h}_i, \overline{\boldsymbol{h}}_\ell \rangle - \langle \boldsymbol{h}_i, \boldsymbol{e}_i \rangle + \langle \boldsymbol{h}_i, \boldsymbol{e}'_i \rangle \\ &= \langle \boldsymbol{h}_i, \overline{\boldsymbol{h}}_\ell \rangle - \mathsf{w}(\boldsymbol{e}_i) + \mathsf{w}(\boldsymbol{e}'_i) \, ,\end{aligned}$$

where the penultimate equality follows from the linearity of the inner product, and the last equality follows from the fact that $\boldsymbol{h}_i$ covers both $\boldsymbol{e}_i$ and $\boldsymbol{e}'_i$.

We distinguish between two cases.

Case 1: $i = \ell$. Here,

$$\langle \boldsymbol{h}_\ell, \overline{\boldsymbol{a}}_i \rangle = \underbrace{\langle \boldsymbol{h}_\ell, \overline{\boldsymbol{h}}_\ell \rangle}_{0} - \underbrace{\mathsf{w}(\boldsymbol{e}_\ell)}_{0} + \mathsf{w}(\boldsymbol{e}'_\ell) = \mathsf{w}(\boldsymbol{e}'_\ell) \le \vartheta - t_\ell \, ,$$

where the last inequality follows from condition (i) of the theorem. Recalling that $\vartheta - t_\ell$ is now the effective threshold of the gate at memory line ℓ (due to the defects of type `N-short`), we therefore conclude from (2) that line ℓ will be selected.

Case 2: $i \ne \ell$. Here,

$$\langle \boldsymbol{h}_i, \overline{\boldsymbol{a}}_i \rangle = \underbrace{\langle \boldsymbol{h}_i, \overline{\boldsymbol{h}}_\ell \rangle}_{\ge \mathsf{d}(\mathcal{C})} - \mathsf{w}(\boldsymbol{e}_i) + \mathsf{w}(\boldsymbol{e}'_i) \ge \mathsf{d}(\mathcal{C}) - \mathsf{w}(\boldsymbol{e}_i) > \vartheta - t_i \, ,$$

where the last inequality follows from condition (ii). We therefore conclude from (2) that line i will be deselected. □

2.5. Discussion

Remark 2.1. Theorem 2.2 holds even when the defects are *temporal* (or *transient*): the values r_i, t_i, and s_i may change during the operation of the demultiplexer, but the demultiplexer will still function properly as long as at each application of the demultiplexer, conditions (i)–(ii) hold. Hence, in practice, our defect models can represent either permanent defects arising during manufacturing, or permanent failures which arise later during operation (in formerly good components), or transient faults in the circuitry. □

Remark 2.2. As an interesting special case of a temporal defect, we can model as such a scenario where an existing input bit to a ϑ-threshold gate is fed by an arbitrary sequence of bits—e.g., a unwanted random sequence (this could happen if, for example, one of the entries in $\boldsymbol{a}$ has been computed incorrectly by the unit that drives the demultiplexer). Here, when an input bit that should have been 1 (respectively, 0) is flipped, we can regard it as a (transient) defect of type `S-closed` (respectively, `S-open`).

As a second scenario, consider a failure where a new input bit is added to a threshold gate and is fed by an arbitrary (unwanted) sequence. When the fed bit is 0, we can regard this failure as a defect of type `N-short` (if the bit is 1, it has no effect on the output of the gate). □

Remark 2.3. Observe that t_i appears with a negative sign in the left-hand side of condition (ii) in Theorem 2.2, which means that defects of type `N-short` "help" to offset defects of type `S-open`. However, condition (ii) holds only if t_i is the *exact* number of defects of type `N-short`, and not just an *upper bound* on that number. If only an upper bound is available‖ (or assumed), then that condition should be changed into:

(ii') $r_i < \mathsf{d}(\mathcal{C}) - \vartheta$. □

As a converse to Theorem 2.2, we show in the Appendix that conditions (i) and (ii) in the theorem are tight in a worst-case sense. We also demonstrate in the Appendix how, under a certain probability model on the occurrence of defects, one can compute the probability that at least one the conditions of Theorem 2.2 is not met, in which case the demultiplexer may fail. The formula given therein can serve as a tool to measure whether the chosen set $\mathcal{C}$ of address configurations and the choice of the threshold ϑ meets the defect-tolerance specifications of the demultiplexer.

The result in [12, Proposition 1] (see also [13]) can be seen as a special case of Theorem 2.2, where the defects are of type `S-open` only and are all aligned across memory lines (i.e., the parameter of interest is the number of address lines that are incident with the defects).

3. Realization using MOS logic

Figure 2 depicts a (somewhat simplified) realization of the abstract threshold-based demultiplexer of Figure 1, using MOS transistors.

‖ For example, consider the second scenario described in Remark 2.2 (of a new input bit that is added to a threshold gate and is fed by an arbitrary sequence). If we let t_i denote the number of failures (regardless of the value of the fed bit), we only get an upper bound on the number of defects of type `N-short`.

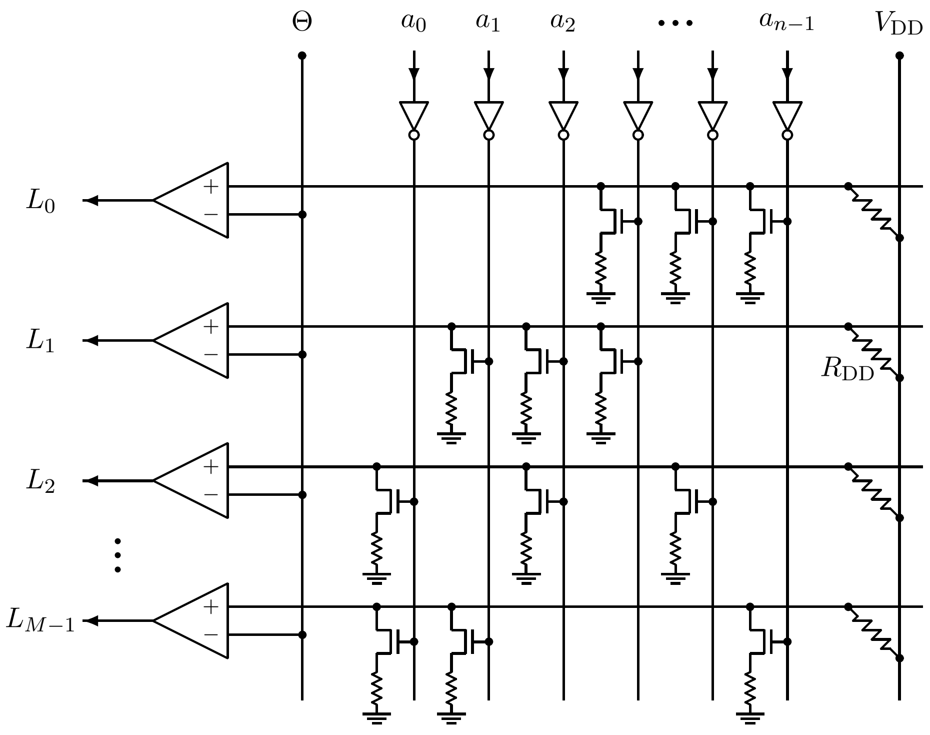


Figure 2. Realization of a demultiplexer using n-MOS transistors.

3.1. Components of the realization

The input word $\boldsymbol{a}$ is fed into the n address lines via inverters, with 1 and 0 being realized as "high" and "low", respectively. There are M cross-wires, each corresponding to a memory line, which are connected to the power supply V_{DD} via pull-up resistors with resistance R_{DD}. Address lines are connected to the cross-wires through n-MOS transistors which have the following characteristic (see Figure 3): when the Gate-to-Source voltage drop V_{GS} is above a certain threshold, the transistor is said to be "closed" and has resistance R_{S}, and when that voltage drop is below that threshold, the transistor is "open" and has infinite resistance. A junction between address line j and cross-wire i is configured with a transistor if, in Figure 1, address line j is connected (through a ϑ-threshold gate) to memory line i (in other words, the junction is configured when $h_{i,j} = 1$).

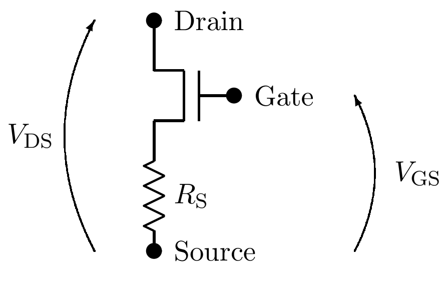


Figure 3. Model of an n-MOS transistor.

The output of memory line i in Figure 3 is given by the output of a voltage comparator, which is fed (through its terminal marked "–") with a reference voltage Θ: the output of the comparator is "high" whenever the voltage level at cross-wire i (measured at the terminal marked "+") equals or exceeds the reference voltage Θ. There are quite a few methods known for implementing voltage comparators—some requiring only four transistors: see [2], [11], [28], [29], and Remark 3.1 below.

The reference voltage Θ, in turn, should be in the range

$$\frac{V_{\mathrm{DD}}}{1+(R_{\mathrm{DD}}/R_{\mathrm{S}})(\vartheta+1)} < \Theta \le \frac{V_{\mathrm{DD}}}{1+(R_{\mathrm{DD}}/R_{\mathrm{S}})\vartheta} . \tag{5}$$

The threshold ϑ can then be expressed as a function of the reference voltage Θ by

$$\vartheta = \left\lfloor \frac{R_{\mathrm{S}}}{R_{\mathrm{DD}}} \left(\frac{V_{\mathrm{DD}}}{\Theta} - 1 \right) \right\rfloor . \tag{6}$$

Next, we verify that at each memory line, the combination of n-MOS transistors and comparators, with the indicated reference voltage Θ, implements properly a ϑ-threshold gate. Suppose that the input word $\boldsymbol{a}$ at the address lines is such that, among all the transistors that are connected to cross-wire i, there are exactly τ transistors which are "closed": this means that $\langle \boldsymbol{h}_i, \overline{\boldsymbol{a}} \rangle = \tau$. These transistors then form τ parallel resistors, each having resistance R_{S} and all connected in series with the pull-up resistor R_{DD}. Cross-wire i therefore acts as the voltage divider shown in Figure 4 and, so, the voltage level at that cross-wire is given by

$$V(\tau) = \frac{V_{\mathrm{DD}}}{1+(R_{\mathrm{DD}}/R_{\mathrm{S}})\tau}$$

(assuming that all resistors are linear). It readily follows from (5) that

$$V(\tau) \ge \Theta \quad \Longleftrightarrow \quad \tau \le \vartheta ,$$

namely, the output L_i in Figure 2 will be "high" precisely when $L_i = 1$ in Figure 1.

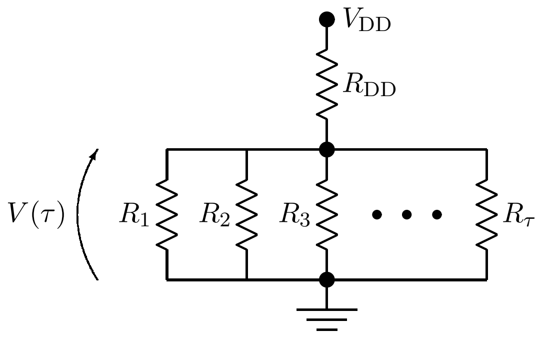


Figure 4. Voltage divider at cross-wire i, where $R_1 = R_2 = \ldots = R_\tau = R_{\mathrm{S}}$.

3.2. Model of defects

In the realization of Figure 2, the types of defects discussed in Section 2.4 can represent the following events:

- Type `S-open` represents the event that a transistor is stuck at "open" (regardless of the voltage V_{GS}). In this case, the respective junction behaves as if the transistor were absent.
- Type `S-closed` represents the event that a transistor is stuck at "closed", and the junction then behaves as if the transistor were replaced by a resistor from Drain to Source with resistance R_{S}.
- Type `N-short` represents the event that in a junction that was not configured with a transistor, the conductance becomes non-negligible (yet the resistance is still at least R_{S}), <u>and</u> the address line that is incident with that junction is at a "low" level¶. Alternatively, type `N-short` can represent a decrease in the effective value of ϑ, as determined by (6).

A decrease in the effective value of ϑ can represent fluctuations in the reference voltage Θ; in addition, we can formally assume *a priori* some reduction of ϑ, for the purpose of increasing the voltage margin of the comparators, as explained next.

3.3. Voltage margin

The *voltage margin* of a comparator in a given circuit is the difference between the following two voltage values:

- The lowest voltage that may appear at the "+" terminal while the output of the comparator should be "high".
- The highest voltage that can appear at the "+" terminal while the output should be "low".

The threshold Θ is then set to be in the range between these two values. A large voltage margin means a more robust operation of the comparator, which explains the desire in practice to increase the voltage margin as much as possible. In the realization of Figure 2, the voltage margin is given by (see (5)):

$$\begin{aligned}\Delta V &= V(\vartheta) - V(\vartheta{+}1)\\ &= \frac{V_{\mathrm{DD}}}{1 + (R_{\mathrm{DD}}/R_{\mathrm{S}})\vartheta} - \frac{V_{\mathrm{DD}}}{1 + (R_{\mathrm{DD}}/R_{\mathrm{S}})(\vartheta{+}1)}\ .\end{aligned}$$

Now, suppose that for some $t > 0$, we require that the number of defects of type `S-closed` in each cross-wire does not exceed $\vartheta - t$, and the number of defects of type `S-open` is less than $\mathsf{d}(\mathcal{C}) - \vartheta$. The parameter t can be seen as an upper bound on the number of "phantom defects" of type `N-short` in each cross-wire, and, in view of

¶ This defect model is therefore temporal in that it depends on the input provided at the address lines (see Remarks 2.1–2.2). While Theorem 2.2 holds for temporal defects as well, the dependence on the contents of common address lines makes such defects statistically dependent across distinct memory lines, and this, in turn, may complicate the computation of the probability of failure of the demultiplexer (see the Appendix). A simple remedy to this is to disregard the contents of the address lines, and consider instead just an *upper bound* on the number of defects of type `N-short`, by counting just the number of non-configured junctions whose conductance becomes non-negligible. When we do this, we should apply Theorem 2.2 with condition (ii') replacing condition (ii).

conditions (i)–(ii') in Theorem 2.2, the demultiplexer will work properly. But reducing[+] the threshold from ϑ to $\vartheta - t$ means, in effect, that we increase the upper boundary in the range (5) from $V(\vartheta)$ into $V(\vartheta{-}t)$, thereby increasing the voltage margin to

$$V(\vartheta{-}t) - V(\vartheta{+}1) .$$

Given ϑ and t ($\le \vartheta$), this difference is maximized when we select R_{S} and R_{DD} so that

$$\frac{R_{\mathrm{S}}}{R_{\mathrm{DD}}} = \sqrt{(\vartheta{+}1)(\vartheta{-}t)} , \tag{7}$$

in which case

$$\begin{aligned} \frac{\Delta V}{V_{\mathrm{DD}}} &= \frac{\left(\sqrt{\vartheta{+}1} - \sqrt{\vartheta{-}t}\right)^2}{t+1} \\ &= \frac{1}{2} \cdot \frac{t+1}{2\vartheta{-}t{+}1} + O\left(\left(\frac{t+1}{2\vartheta{-}t{+}1}\right)^3\right) , \end{aligned} \tag{8}$$

where $O(\cdot)$ stands for an expression that grows at most linearly with its argument. In addition, for the choice (7), we get that the "low" voltage interval $[0, V(\vartheta{+}1)]$ and the "high" voltage interval $[V(\vartheta{-}t), V_{\mathrm{DD}}]$ are of the same size.

Figure 5 illustrates the possible values of the voltage level at any given cross-wire i, as a function of the parameters ϑ, t, and $d = \mathsf{d}(\mathcal{C})$, and the number of defects of type `S-open` and `S-closed`. The voltage range (between 0 and V_{DD}) is sub-divided into several intervals, each corresponding to an event which is characterized by three attributes: the identity of the selected line (whether it is i or not), the number r_i of defects of type `S-open` in cross-wire i, and the number s_i of defects of type `S-closed` in that cross-wire. The output of the comparator at cross-wire i is shown to the right. The point $V(d)$ marks the largest possible voltage level at any deselected cross-wire in case it is defect-free. (The figure is drawn to scale for $\vartheta = 2$, $t = 1$, $d = 4$, and R_{S} and R_{DD} that satisfy (7).)

3.4. Discussion

Remark 3.1. Except for possible fluctuations in the reference voltage or the need for a larger voltage margin, our defect model assumes that the comparators are otherwise reliable (whereas the n-MOS transistors can get stuck "open" or "closed"). We can justify this assumption by the fact that the number of comparators in the realized demultiplexer is considerably smaller than the number of n-MOS transistors; hence, for a given cost, more can be invested in the manufacturing of the comparators than in the transistors.

As a concrete example, Figure 6 presents an implementation of a voltage comparator using four transistors. The implementation consists of two inverter-like CMOS stages, where the threshold voltage is set by adjusting the relative sizes of the two transistors

[+] Our seeming one-sided discussion, which only considers reducing ϑ rather than also increasing it, is rather arbitrary: we could instead set the original threshold to $\vartheta' = \vartheta - t$, and add up to t "phantom defects" of type `S-open`.

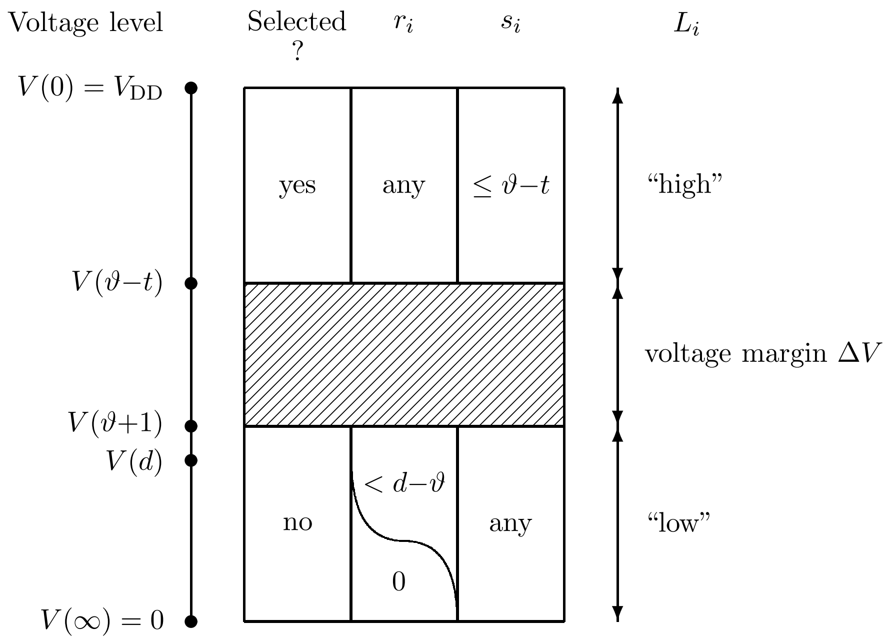


Figure 5. Range of voltage levels at cross-wire i.

in each stage [28]. This comparator would fail if any of the four transistors in Figure 6 gets stuck "open" or "closed"; yet, we can deal with this problem by "hardening" the transistors in the comparator, e.g., by making them larger than the other transistors in the demultiplexer circuit. (A similar approach—of increasing reliability of components that constitute only a small proportion of a circuit—has been suggested elsewhere: see, for example, [22, Section 4.1].) □

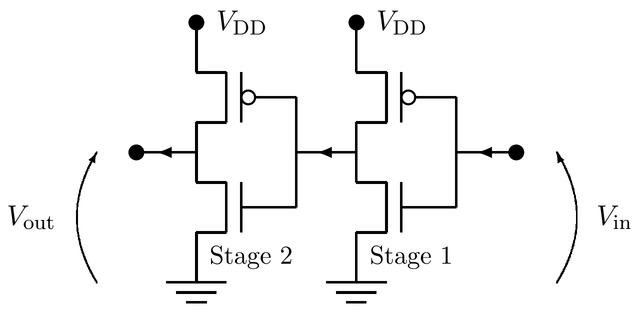


Figure 6. CMOS implementation of a voltage comparator.

Remark 3.2. In practice, a memory system that uses the demultiplexer of Figure 2 as part of its addressing scheme, should also include an *encoder*, which maps, in a one-to-one manner, the raw address of a memory line into the address configuration assigned to that line. Such an encoder can serve multiple memory units and, therefore, its complexity can be amortized over them. The encoder can be implemented using look-up tables; furthermore, some of the constructions of unisymmetric codes were specifically designed to have encoders with relatively simple hardware implementations: see [3], [4], [5], [7], and [23]. □

Example 3.1. Suppose that a demultiplexer is sought with $M = 2^{10}$ memory lines, with defect tolerance of up to one defect of type `S-open` and one defect of type `S-closed` per cross-wire. We will choose the address configurations to belong to an $(n, M{=}2^{10}, d{=}4)$-unisymmetric code and will take $\vartheta = 2$. By Theorem 2.2 it follows that these parameters guarantee the defect-tolerance specifications of the demultiplexer.

We take the unisymmetric code to be in fact a constant-weight code: from [6, Table I-C] we see that there is a known construction of an $(n{=}23, M'{=}1288, 2d{=}8, w{=}11)$-constant-weight code, out of which we take 2^{10} codewords to serve as the address configurations. The number of n-MOS transistors in the demultiplexer (comparators excepted) equals $w \cdot M = 11\,264$, and, after optimizing over the ratio $R_{\rm S}/R_{\rm DD}$, we get from (8) that the voltage margin (relative to $V_{\rm DD}$) of each comparator is

$$\frac{\Delta V}{V_{\rm DD}} = \frac{1}{2} \cdot \left(\sqrt{3} - 1\right)^2 \approx 0.268 \; .$$

Note that for the same value of ϑ we can tolerate two defects of type `S-closed`, at the expense of reducing the relative voltage margin to $\left(\sqrt{3} - \sqrt{2}\right)^2 \approx 0.101$.

We now compare this circuit with the one obtained from the design in [24]. From [1, Table I] we get that an $(\tilde{n}, M{=}2^{10}, 2\tilde{d}{=}4, \tilde{w})$-constant-weight code exists only when $\tilde{n} \geq 16$, and there is a known construction of such a code with $\tilde{n} = 16$ and $\tilde{w} = 8$ [6, Table XV]. We use the codewords of this code as the address configurations in the circuit proposed in [24], resulting in $\tilde{w} \cdot M = 8\,192$ configured junctions. Recall, however, that defects of type `S-closed` are handled in [24] by a series replication of each transistor; hence, this design will require a total of 16 384 transistors* (this number increases to $3 \times 8\,192 = 24\,576$ if two defects of type `S-closed` are to be tolerated per cross-wire). □

Remark 3.3. In Example 3.1, we have constructed our demultiplexer (in Figure 2) taking $d = 4$, whereas for the scheme of [24] it was sufficient to take $\tilde{d} = 2$ in order to tolerate the same number of defects of type `S-open`. The difference $(d - \tilde{d} = 2)$ was "invested" in our scheme in handling defects of type `S-closed` without the need for series replication, and in increasing the voltage margin. Now, it is known that when $d \geq 3$ and n is a prime power, then there is an $(n, M, 2d, w)$-constant-weight code with parameters

$$M \geq \frac{1}{n^{d-1}} \binom{n}{w}$$

* While the transistor count seems to be a predominant parameter to consider when comparing complexities, by no means should it be the only one. An extreme—and quite absurd—demultiplexer could be obtained by taking a $(dM, M, 2d, d)$-constant-weight code in which no two distinct codewords share a 1 at the same position. If we had used this code in our example, we would have reduced the number of transistors by (more than) a factor of 2, yet the number of address lines would have been prohibitively large. In selecting the code parameters, we have taken the parameters n and $\tilde{n}$ to be the smallest that could accommodate the size M and the desired distance properties.

(see [6, Theorem 15]). Taking this lower bound as a guideline for the choice of the parameters $\tilde{n}$ and $\tilde{w}$ (in the scheme of [24]) and of n and w (in our scheme), we see that when the difference $d-\tilde{d}$ remains fixed and M grows, then for virtually all feasible choices for $\tilde{n}$ and $\tilde{w}$, we will be able to select n and w so that the ratios $n/\tilde{n}$ and $w/\tilde{w}$ approach 1. Therefore, asymptotically, avoiding the series replication of [24] will reduce the number of transistors by a factor of 2 (or, generally, by a larger factor if the specifications require to tolerate more than one defect of type `S-open` in each memory line). □

Example 3.2. Continuing Example 3.1, it is interesting to compare our scheme to [24] also under the probability model where a transistor gets stuck "open" (respectively, "closed") with probability p_{o} (respectively, p_{c}), independently of all other transistors, assuming that $p_{\mathsf{o}}, p_{\mathsf{c}} \ll 1/w$ (see also the Appendix). In our scheme, "stuck-open" defects will cause a given cross-wire to fail with probability approximately $\frac{1}{2}w(w-1)p_{\mathsf{o}}^2 \approx (7.4p_{\mathsf{o}})^2$, compared to $2\tilde{w}(\tilde{w}-1)p_{\mathsf{o}}^2 \approx (10.6p_{\mathsf{o}})^2$ in [24]. On the other hand, defects of type "closed" will cause a given cross-wire in [24] to fail with probability $\tilde{w}p_{\mathsf{c}}^2 \approx (2.8p_{\mathsf{c}})^2$, compared to $\frac{1}{2}w(w-1)p_{\mathsf{c}}^2 \approx (7.4p_{\mathsf{c}})^2$ in our scheme if only one defect of type `S-closed` is tolerated per cross-wire, or to $\frac{1}{6}w(w-1)(w-2)p_{\mathsf{c}}^3 \approx (5.5p_{\mathsf{c}})^3$ if two such defects are tolerated.

Figure 7 depicts (in logarithmic scale) the probability $\mathsf{P}_{\mathrm{fail}} = \mathsf{P}_{\mathrm{fail}}(p_{\mathsf{o}}, p_{\mathsf{c}})$ of a failure event in a demultiplexer with $M = 2^{10}$ memory lines, under several design strategies (by a "failure event" we mean that at least one memory line in the whole demultiplexer can be either incorrectly selected or incorrectly deselected). The curve marked as "Sperner set" corresponds to the design described in Section 2.1, which offers essentially no defect tolerance: the address configurations are taken from a Sperner set of length $n = 13$, which is the shortest Sperner set that contains at least 2^{10} words. The curve marked as "Replication" corresponds to the scheme of [24], using a $(16, 2^{10}, 4, 8)$-constant-weight code and two-fold series replication. The remaining two curves show the performance of our design, with threshold value $\vartheta = 2$ and with $\mathcal{C}$ taken as a $(23, 2^{10}, 8, 11)$-constant-weight code. In one curve, it is assumed that the comparators require an increase of the voltage margin by selecting $t = 1$ in (8): this curve lies above the "Replication" curve when p_{c} gets large compared to p_{o}. The curves were computed using the analysis in the Appendix. □

4. Variant using resistor logic

In this section, we present a realization of a threshold-based demultiplexer using resistor logic. Our motivation to consider such demultiplexers is that they may lend themselves more easily to a nano-scale implementation. As we see, the demultiplexer presented here behaves somewhat differently than the demultiplexers in Figure 1 or Figure 2.

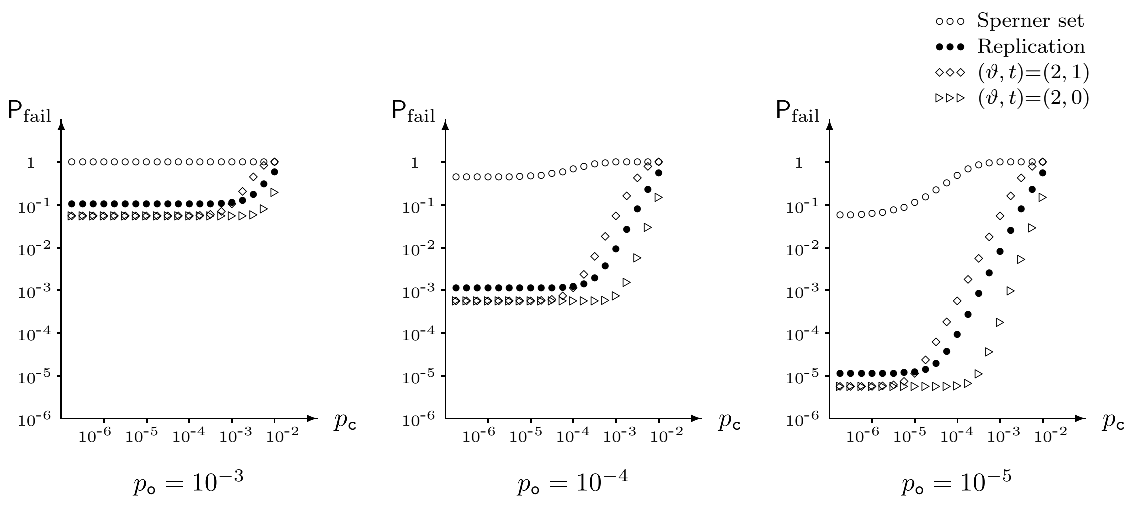


Figure 7. Probability of failure of a demultiplexer with 2^{10} memory lines.

4.1. Components of the realization

Figure 8 presents a resistor-logic realization of a demultiplexer. This figure is similar

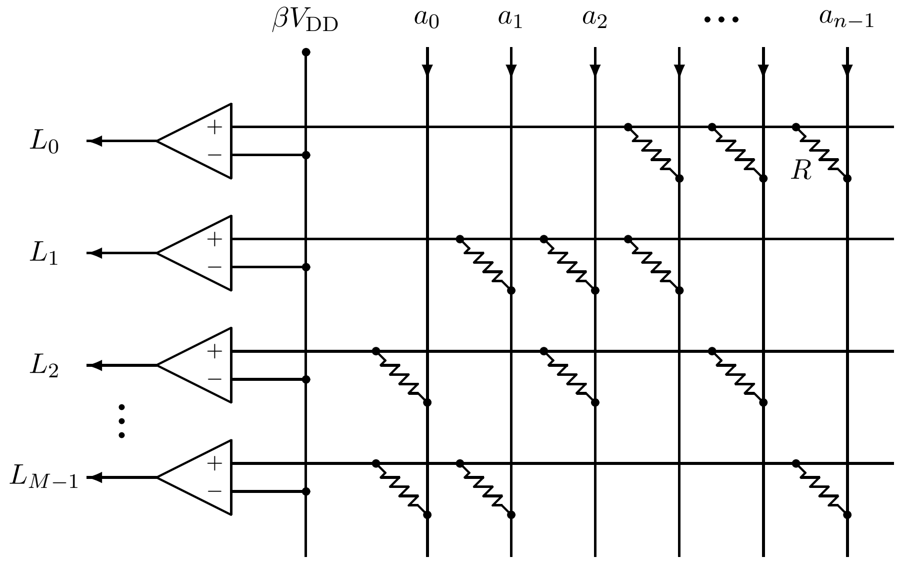


Figure 8. Realization of a demultiplexer using resistor logic.

to the MOS realization of Figure 2, except for the following changes: (a) instead of the n-MOS transistors, the junctions are now configured with resistors—all assumed to have the same resistance R; (b) no inversion is applied to the bits of the input word $\boldsymbol{a}$; and (c) the cross-wires are not pulled up to V_{DD}.

The n address lines are fed with the input word $\boldsymbol{a}$, with 1 and 0 being realized, respectively, as V_{DD} (“high”) and ground (“low”), and the reference voltage is set to

βV_{DD} for some real β in the range $0 \le \beta \le 1$.

Remark 4.1. One potential application of the design presented here is the addressing of a nano-scale memory from a micro-scale circuit. In such a setting, the address lines in Figure 8 will have micro-scale dimensions, whereas the cross wires, the voltage comparators (which realize the threshold gates in the demultiplexer), and the memory lines will be in the nano-scale level. The defects will occur in the junctions, namely, in the interface between the two scale levels. In Section 4.4, we will show a possible nano-scale implementation of comparators using hysteretic resistors [25], [26]. □

Next, we analyze the defect tolerance of the circuit, in terms of β and the set of address configurations.

Suppose first that the demultiplexer is defect-free, and consider any cross-wire i. That wire is connected to $\mathsf{w}(\boldsymbol{h}_i)$ resistors, out of which $\sigma = \langle \boldsymbol{h}_i, \boldsymbol{a} \rangle$ pull up the wire to V_{DD}, whereas the remaining $\tau = \langle \boldsymbol{h}_i, \overline{\boldsymbol{a}} \rangle$ resistors pull it down to ground. Therefore, cross-wire i acts as the voltage divider shown in Figure 9, and the voltage level (relative to V_{DD}) at that wire is

$$\frac{V(\sigma,\tau)}{V_{\mathrm{DD}}} = \frac{R/\tau}{(R/\tau)+(R/\sigma)} = \frac{\sigma}{\sigma+\tau} = \frac{\langle \boldsymbol{h}_i, \boldsymbol{a} \rangle}{\mathsf{w}(\boldsymbol{h}_i)} \,. \tag{9}$$

Thus,

$$L_i = 1 \quad \Longleftrightarrow \quad \frac{\langle \boldsymbol{h}_i, \boldsymbol{a} \rangle}{\mathsf{w}(\boldsymbol{h}_i)} \ge \beta \tag{10}$$

(so, as opposed to (1), it is now the ratio—rather than the difference—between $\langle \boldsymbol{h}_i, \boldsymbol{a} \rangle$ and $\mathsf{w}(\boldsymbol{h}_i)$ that is compared to the threshold). Defining $\gamma = 1 - \beta$, we can rewrite (10) as

$$L_i = 1 \quad \Longleftrightarrow \quad \frac{\langle \boldsymbol{h}_i, \overline{\boldsymbol{a}} \rangle}{\mathsf{w}(\boldsymbol{h}_i)} \le \gamma \,. \tag{11}$$

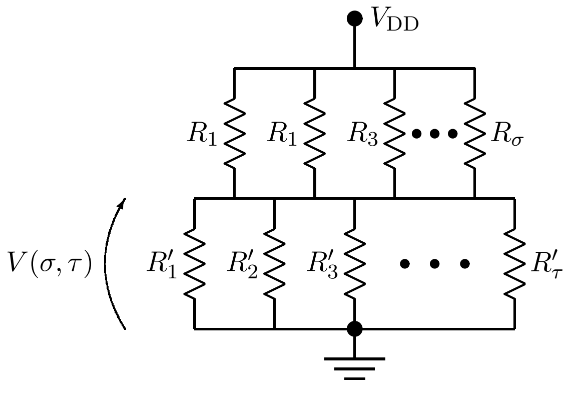


Figure 9. Voltage divider (all resistors are the same).

Let $\mathcal{C} = \{\boldsymbol{h}_0, \boldsymbol{h}_1, \ldots, \boldsymbol{h}_{M-1}\}$ be the set of address configurations in the demultiplexer in Figure 8. As was the case in the previous demultiplexers, we select memory line ℓ by setting the input word $\boldsymbol{a}$ to $\boldsymbol{h}_\ell$. We have $\langle \boldsymbol{h}_i, \overline{\boldsymbol{h}_\ell} \rangle = 0$ for $i = \ell$, and $\langle \boldsymbol{h}_i, \overline{\boldsymbol{h}_\ell} \rangle \geq \mathsf{d}(\mathcal{C})$ for $i \neq \ell$. It follows from (10)–(11) that the demultiplexer in Figure 8 will uniquely select any given memory line if $\mathcal{C}$ and γ are related by

$$\mathsf{d}(\mathcal{C}) > \gamma \cdot \max_{\boldsymbol{h} \in \mathcal{C}} \mathsf{w}(\boldsymbol{h}) \ .$$

4.2. Model of defects

Next, we consider the tolerance of the demultiplexer in Figure 8 under the following two types of defects:

Resistor open (`R-open`): a junction which was configured with a resistor becomes disconnected, thereby increasing the resistance from R to ∞.

Resistor short (`R-short`): a junction which was not configured with a resistor becomes a conductor, yet its resistance is still at least R.

(One could regard the non-configured junctions as if they are "configured" with resistors of infinite resistance; from such a standpoint, defects of type `R-short` could also be called `R-closed`, in analogy with the respective defects defined in Section 2.4.)

We have the following result.

Theorem 4.1. *Let* $\mathcal{C} = \{\boldsymbol{h}_0, \boldsymbol{h}_1, \ldots, \boldsymbol{h}_{M-1}\}$ *be the set of address configurations assigned to the* M *address lines of a given demultiplexer as in Figure 8, and write* $w_i = \mathsf{w}(\boldsymbol{h}_i)$. *Let* βV_{DD} *be the reference voltage fed into each comparator, where* $0 \leq \beta \leq 1$, *and define* $\gamma = 1 - \beta$. *For every* i *in the range* $0 \leq i < M$, *denote by* r_i *and* t_i *the number of defects of type* `R-open` *and* `R-short`, *respectively, at cross-wire* i, *and suppose that for every* i, *the following two conditions hold:*

(A) $\gamma r_i + \beta t_i \leq \gamma w_i$.

(B) $\beta r_i + \gamma t_i < \mathsf{d}(\mathcal{C}) - \gamma w_i$.

Then for every ℓ *in the range* $0 \leq \ell < M$, *setting the input word* $\boldsymbol{a}$ *to* $\boldsymbol{h}_\ell$ *will uniquely select memory line* ℓ.

Proof. First, observe that conditions (A) and (B) are equivalent, respectively, to

$$1 - \frac{t_i}{w_i - r_i + t_i} \geq \beta \tag{12}$$

and

$$1 - \frac{\mathsf{d}(\mathcal{C}) - r_i}{w_i - r_i + t_i} < \beta \ . \tag{13}$$

In what follows, we will show that the left-hand side of (12) is the voltage level (relative to V_{DD}) at cross-wire i when memory line i is selected, and the left-hand side of (13) is an upper bound on the relative voltage level at cross-wire i when memory line i is deselected.

Suppose that memory line ℓ is to be selected, in which case $\boldsymbol{a} = \boldsymbol{h}_\ell$. Also assume for the time being that each defect of type `R-short` reduces the resistance in the non-configured junction all the way down to R.

For each i, the effect of each defect at cross-wire i is to flip an entry in $\boldsymbol{h}_i$: defects of type `R-open` change 1 into 0, whereas defects of type `R-short` change 0 into 1. The resulting configuration at cross-wire i can be written as

$$\boldsymbol{h}_i^* = \boldsymbol{h}_i - \boldsymbol{e}_i + \boldsymbol{e}_i' \; , \tag{14}$$

where $\boldsymbol{e}_i$ (respectively, $\boldsymbol{e}_i'$) is a word in $\{0,1\}^n$ which contains 1 where the defects of type `R-open` (respectively, `R-short`) flip entries in $\boldsymbol{h}_i$; note that $\boldsymbol{e}_i \subseteq \boldsymbol{h}_i$ and $\boldsymbol{e}_i' \subseteq \overline{\boldsymbol{h}}_i$. We have

$$\mathsf{w}(\boldsymbol{h}_i^*) = \mathsf{w}(\boldsymbol{h}_i) - \mathsf{w}(\boldsymbol{e}_i) + \mathsf{w}(\boldsymbol{e}_i') = w_i - r_i + t_i \tag{15}$$

and

$$\langle \boldsymbol{h}_i^*, \overline{\boldsymbol{a}} \rangle = \langle \boldsymbol{h}_i{-}\boldsymbol{e}_i{+}\boldsymbol{e}_i', \overline{\boldsymbol{h}}_\ell \rangle = \langle \boldsymbol{h}_i, \overline{\boldsymbol{h}}_\ell \rangle - \langle \boldsymbol{e}_i, \overline{\boldsymbol{h}}_\ell \rangle + \langle \boldsymbol{e}_i', \overline{\boldsymbol{h}}_\ell \rangle \; .$$

We distinguish between two cases.

Case 1: $i = \ell$. Here,

$$\langle \boldsymbol{h}_\ell^*, \overline{\boldsymbol{a}} \rangle = \underbrace{\langle \boldsymbol{h}_\ell, \overline{\boldsymbol{h}}_\ell \rangle}_{0} - \underbrace{\langle \boldsymbol{e}_\ell, \overline{\boldsymbol{h}}_\ell \rangle}_{0} + \underbrace{\langle \boldsymbol{e}_\ell', \overline{\boldsymbol{h}}_\ell \rangle}_{\mathsf{w}(\boldsymbol{e}_\ell')} = t_\ell$$

and, so, by (9), the voltage level (relative to V_{DD}) at cross-wire ℓ equals

$$\frac{\langle \boldsymbol{h}_\ell^*, \boldsymbol{a} \rangle}{\mathsf{w}(\boldsymbol{h}_\ell^*)} = 1 - \frac{\langle \boldsymbol{h}_\ell^*, \overline{\boldsymbol{a}} \rangle}{\mathsf{w}(\boldsymbol{h}_\ell^*)} = 1 - \frac{t_\ell}{w_\ell{-}r_\ell{+}t_\ell} \; .$$

Thus, by (12) we deduce that that memory line ℓ will be selected. This will also be the case if some of the t_ℓ resistors that are introduced in cross-wire ℓ by defects of type `R-short` have resistance strictly greater than R: this will translate into increasing the resistances of the respective resistors R_x' in the voltage divider in Figure 9, thereby only increasing the voltage level $V(\sigma, \tau)$ at cross-wire ℓ.

Case 2: $i \neq \ell$. Here,

$$\langle \boldsymbol{h}_i^*, \overline{\boldsymbol{a}} \rangle = \underbrace{\langle \boldsymbol{h}_i, \overline{\boldsymbol{h}}_\ell \rangle}_{\geq \mathsf{d}(\mathcal{C})} - \underbrace{\langle \boldsymbol{e}_i, \overline{\boldsymbol{h}}_\ell \rangle}_{\leq \mathsf{w}(\boldsymbol{e}_i)} + \underbrace{\langle \boldsymbol{e}_i', \overline{\boldsymbol{h}}_\ell \rangle}_{\geq 0} \geq \mathsf{d}(\mathcal{C}) - r_i$$

and, so, the relative voltage level at cross-wire i can be bounded from above by

$$\frac{\langle \boldsymbol{h}_i^*, \boldsymbol{a} \rangle}{\mathsf{w}(\boldsymbol{h}_i^*)} = 1 - \frac{\langle \boldsymbol{h}_i^*, \overline{\boldsymbol{a}} \rangle}{\mathsf{w}(\boldsymbol{h}_i^*)} \leq 1 - \frac{\mathsf{d}(\mathcal{C}) - r_i}{w_i{-}r_i{+}t_i} \; .$$

Hence, by (13) we conclude that memory line i will be deselected. Notice that this holds even in the worse-case scenario where $\langle \boldsymbol{e}_i', \overline{\boldsymbol{h}}_\ell \rangle = 0$, namely, when all the t_i resistors added by defects of type `R-short` (have resistance R and) pull cross-wire i up to V_{DD}. If some of these resistors have resistance greater than R, then the voltage level at cross-wire i will only decrease, and it will certainly do so if any of those added resistors pull cross-wire i down to ground instead of pulling it up. □

In what follows, we analyze the shape of the set of pairs (r_i, t_i) that satisfy conditions (A)–(B) in Theorem 4.1, assuming that the set $\mathcal{C}$ of address configurations forms an $(n, M, 2d, w)$-constant-weight code (where $d = \mathsf{d}(\mathcal{C})$); thus, $\mathsf{w}(\boldsymbol{h}_i) = w_i = w$ for all i and, as in every constant-weight code, we must also have $d \leq w$. Defining ϑ to be the real value γw, we can rewrite conditions (A)–(B) in the following form (omitting the subscripts from r_i and t_i for simplicity):

(A') $\gamma r + \beta t \leq \vartheta$.
(B') $\beta r + \gamma t < d - \vartheta$.

We let $\mathcal{T} = \mathcal{T}(\beta, w, d)$ be the set of all pairs (r, t) of nonnegative integers that satisfy conditions (A')–(B'). Equivalently,

$$\mathcal{T} = \left\{ (r,t) \in \mathbb{N} \times \mathbb{N} \, : \, \frac{t}{w{-}r{+}t} \leq 1{-}\beta < \frac{d{-}r}{w{-}r{+}t} \right\} , \tag{16}$$

where $\mathbb{N}$ denotes the set of nonnegative integers.

Regarding r and t momentarily as real-valued, each of the two conditions (A') and (B') defines a region in the shape of a right-angled triangle, in the first quadrant of the (r,t)-plane: the two legs of the triangle are formed by the axes, the hypotenuse in the case of condition (A') passes through the points $(w{=}\vartheta/\gamma, 0)$ and $(0, \vartheta/\beta)$, and in the case of condition (B') it passes through $((d{-}\vartheta)/\beta, 0)$ and $(0, (d{-}\vartheta)/\gamma)$. The inequality $d \leq w$ implies that the point $(\vartheta/\gamma, 0)$ is never to the left of $((d{-}\vartheta)/\beta, 0)$.

When $\beta \leq w/(w{+}d)$, the point $(0, \vartheta/\beta)$ lies above or coincides with $(0, (d{-}\vartheta)/\gamma)$, in which case condition (A') is implied by condition (B'). Figure 10a shows the set $\mathcal{T}$ in this case, marked by the (filled) bullets in the figure and bounded by the dotted line. The (shade-margined) double line marks the boundary of the nonnegative real points (r,t) that satisfy condition (B') (the boundary points themselves do not satisfy the condition). The set $\mathcal{T}$ in Figure 10a becomes empty when $\beta \leq (w{-}d)/w$: we then get $\vartheta \geq d$.

When $w/(w{+}d) < \beta \leq 1$, the two right-angled triangles that correspond to the two conditions intersect only partially, thereby defining the region shown in Figure 10b: the two shade-margined segments of the boundary intersect at the point $(\beta d{-}\vartheta, \vartheta{-}\gamma d)/(\beta{-}\gamma)$ (integer boundary points belong to $\mathcal{T}$ unless they are on the double-lined segment). When $\beta \to 1$, the set $\mathcal{T}$ becomes a rectangle and, interestingly, it coincides then with the region defined by conditions (i)–(ii') of Theorem 2.2 (assuming $s_i = 0$)♯.

Example 4.1. We consider the demultiplexer of Figure 8, with $\beta = \frac{5}{6}$ and the address configurations forming an $(n{=}23, M{=}2^{10}, 2d{=}8, w{=}11)$-constant-weight code (see Example 3.1). After clearing denominators, conditions (A')–(B') take the form

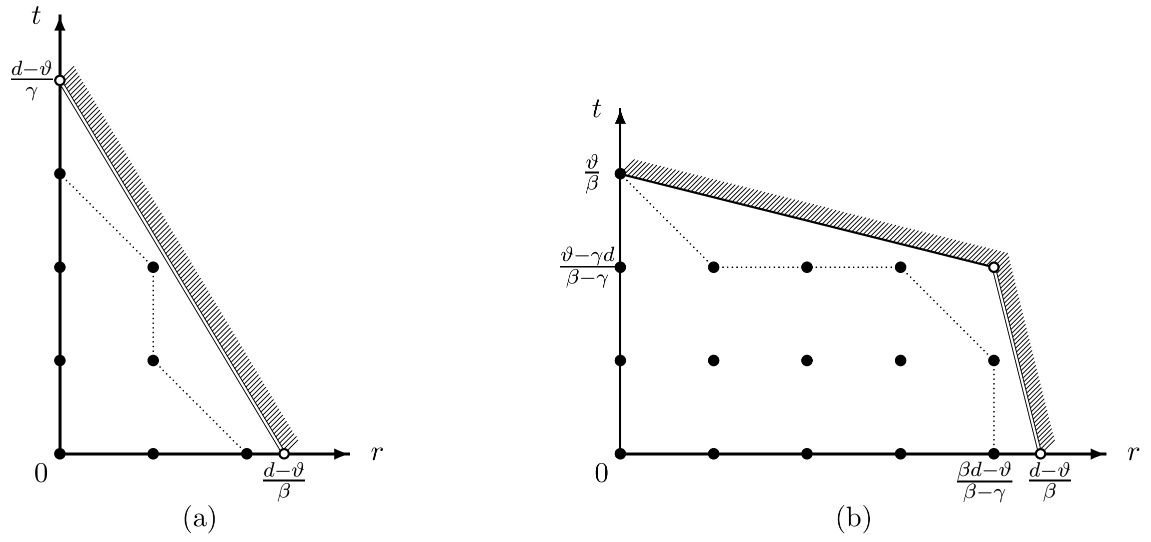
(A') $r + 5t \leq 11$.
(B') $5r + t < 13$.

The set $\mathcal{T}$ in this case has size 8 and is given by

$$\mathcal{T} = \mathcal{T}(\tfrac{5}{6}, 4, 11) = \\ \Big\{ (0,0), (0,1), (0,2), (1,0), (1,1), (1,2), (2,0), (2,1) \Big\}$$

(see Figure 11a). □

♯ Note however that if ϑ is to remain bounded away from 0 while $\beta \to 1$, then w needs to go to infinity.



t
$\frac{d-\vartheta}{\gamma}$
0
$\frac{d-\vartheta}{\beta}$
r
(a)
t
$\frac{\vartheta}{\beta}$
$\frac{\vartheta-\gamma d}{\beta-\gamma}$
0
$\frac{\beta d-\vartheta}{\beta-\gamma}$
$\frac{d-\vartheta}{\beta}$
r
(b)

Figure 10. (a) Set $\mathcal{T}(\beta, w, d)$ when $\beta \leq w/(w+d)$. (b) Set $\mathcal{T}(\beta, d, w)$ when $\beta > w/(w+d)$.

4.3. Voltage margin

Consider the demultiplexer in Figure 8 when the set of address configurations form an $(n, M, 2d, w)$-constant-weight code, and let $\mathcal{T} = \mathcal{T}(\beta, w, d)$ be as in (16). Define the rational numbers γ_{high} and γ_{low} by

$$\gamma_{\mathrm{high}} = \max_{(r,t)\in\mathcal{T}} \frac{t}{w-r+t} \qquad \text{and} \qquad \gamma_{\mathrm{low}} = \min_{(r,t)\in\mathcal{T}} \frac{d-r}{w-r+t} .$$

From (16) we have $\gamma_{\mathrm{high}} \leq 1-\beta < \gamma_{\mathrm{low}}$.

Suppose that the numbers r_i and t_i of defects satisfy conditions (A')–(B') (namely, $(r_i, t_i) \in \mathcal{T}$). Recall that the left-hand side of (12) is the relative voltage level at cross-wire i when memory line i is selected; by the definition of γ_{high}, that relative voltage level is bounded from below by $1-\gamma_{\mathrm{high}}$. Similarly, the left-hand side of (13) is an upper bound on the relative voltage level at cross-wire i when memory line i is deselected; that relative voltage level is therefore bounded from above by $1-\gamma_{\mathrm{low}}$. We conclude that the relative voltage margin of the comparator at any memory line is bounded from below by

$$\gamma_{\mathrm{low}} - \gamma_{\mathrm{high}} .$$

As demonstrated in the next example, we can increase this margin by pruning the set $\mathcal{T}$, namely, by assuming stronger conditions than (A')–(B').

Example 4.2. For the demultiplexer of Example 4.1, we get

$$\gamma_{\mathrm{high}} = \max_{(r,t)\in\mathcal{T}} \frac{t}{11-r+t} = \left.\frac{t}{11-r+t}\right|_{(r,t)=(1,2)} = \frac{1}{6}$$

and

$$\gamma_{\mathrm{low}} = \min_{(r,t)\in\mathcal{T}} \frac{4-r}{11-r+t} = \left.\frac{4-r}{11-r+t}\right|_{(r,t)=(2,1)} = \frac{1}{5} ,$$

thereby yielding the following lower bound on the relative voltage margin:

$$\gamma_{\text{low}} - \gamma_{\text{high}} = \frac{1}{30} \approx 0.033$$

(this margin is too small for any practical purposes). However, suppose that we only expect to have at most one defect of each type (`R-open` and `R-short`) in each cross-wire. This allows us to prune $\mathcal{T}$ into

$$\mathcal{T}' = \Big\{(0,0),(0,1),(1,0),(1,1)\Big\}$$

(see Figure 11b), for which we get

$$\gamma'_{\text{high}} = \max_{(r,t)\in\mathcal{T}'} \frac{t}{11-r+t} = \frac{1}{11}$$

and

$$\gamma'_{\text{low}} = \min_{(r,t)\in\mathcal{T}'} \frac{4-r}{11-r+t} = \frac{3}{11}$$

(the maximum and minimum are both attained at $(r,t) = (1,1)$). Thus, the pruning increases the lower bound on the relative voltage margin considerably to:

$$\gamma'_{\text{low}} - \gamma'_{\text{high}} = \frac{2}{11} \approx 0.182 \; .$$

This lower bound increases even further if we can assume in addition that defects of type `R-open` and `R-short` cannot occur simultaneously at the same cross-wire. In this case, we can prune $\mathcal{T}$ into

$$\mathcal{T}'' = \Big\{(0,0),(0,1),(1,0)\Big\}$$

(see Figure 11c) and get

$$\gamma''_{\text{high}} = \max_{(r,t)\in\mathcal{T}''} \frac{t}{11-r+t} = \frac{1}{12}$$

and

$$\gamma''_{\text{low}} = \min_{(r,t)\in\mathcal{T}''} \frac{4-r}{11-r+t} = \frac{3}{10} \; ,$$

thereby yielding

$$\gamma''_{\text{low}} - \gamma''_{\text{high}} = \frac{13}{60} \approx 0.217 \; .$$

□

4.4. Implementation of comparators using hysteretic resistors

Borrowing ideas from [25] and [26], we describe here a possible implementation of voltage comparators using hysteretic resistors, as such an implementation may be suitable for nano-scale applications.

We model a hysteretic resistor as a two-terminal electronic device which can be in one of two states, "on" and "off". Depending on the state it is in, the resistance of the device takes one of two values, R_{on} or R_{off}, where $R_{\text{on}} \ll R_{\text{off}}$ (e.g., the values assumed

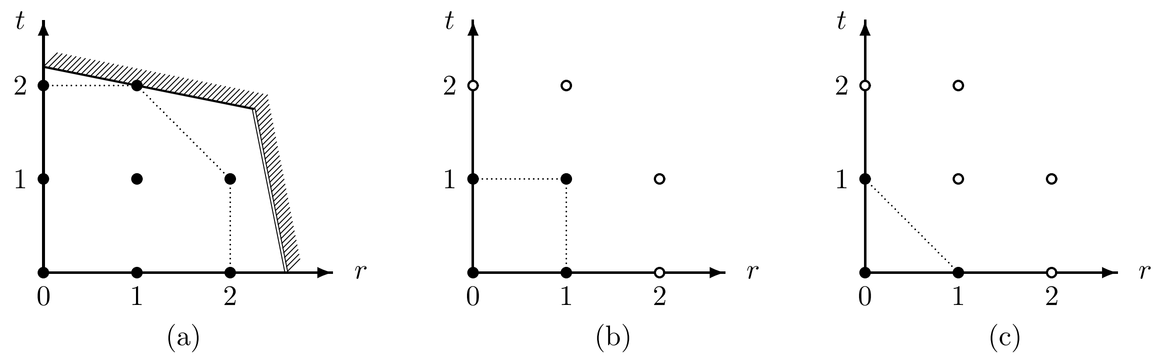


Figure 11. (a) Set $\mathcal{T}$ for Example 4.1. (b) Pruned set $\mathcal{T}'$ for Example 4.2. (c) Pruned set $\mathcal{T}''$ for Example 4.2.

in [25] are $R_{\text{on}} = 10^6\Omega$ and $R_{\text{off}} = 10^9\Omega$). The device remains in its state as long as the voltage drop across it is within the interval $[V_{\text{off}}, V_{\text{on}}]$, where V_{on} (respectively, V_{off}) is a prescribed positive (respectively, negative) voltage value. When the voltage drop across the device exceeds V_{on} then it switches to "on"; otherwise, a voltage drop below V_{off} switches the device to "off'.

Figure 12 shows an implementation of a voltage comparator using a hysteretic resistor, where the reference voltage is fixed to V_{on}. The resistance R_{g} is selected so that $R_{\text{on}} \ll R_{\text{g}} \ll R_{\text{off}}$ (e.g., for the resistance values assumed in [25], we can take R_{g} to be of the order $10^7\Omega$). The comparator is operated in two phases. In the first phase, the hysteretic resistor is switched to "off" by setting V_{in} to a voltage level lower than V_{off} (the diode prevents destruction of the hysteretic resistor in this phase—see [25]). In the second phase, V_{in} is set to the compared (input) voltage level. If $V_{\text{in}} \le V_{\text{on}}$ then the hysteretic resistor remains in its "off" state and, therefore, the output voltage will satisfy $V_{\text{out}} \le V_{\text{on}} R_{\text{g}}/(R_{\text{g}}+R_{\text{off}})$; and under our assumption that $R_{\text{g}} \ll R_{\text{off}}$, this voltage will be close to ground. On the other hand, if $V_{\text{in}} > V_{\text{on}}$ then the hysteretic resistor switches to "on", in which case the output voltage will satisfy $V_{\text{out}} > V_{\text{on}} R_{\text{g}}/(R_{\text{g}}+R_{\text{on}}) \approx V_{\text{on}}$.

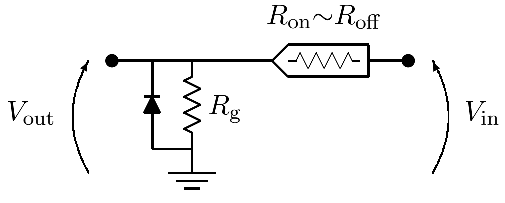


Figure 12. Implementation of a voltage comparator using a hysteretic resistor.

The implementation of Figure 12 can now be incorporated into the design in Figure 8 by setting V_{DD} to V_{on}/β. In addition, R_{g} should be relatively large compared to the resistance R of a configured junction. In the first phase, all address lines are fed with a negative voltage so that the hysteretic resistors are switched to "off". In the second phase, the address lines are fed with the input word $\boldsymbol{a}$ (with 1 and 0 being realized, respectively, as V_{DD} and ground). As long as the defect count is within the designed tolerance, the comparator of the selected line will be the only one to switch to "on" and (assuming that $R \ll R_{\mathrm{g}}$), the output voltage V_{out} of that comparator will be approximately V_{on}. The other comparators, all remaining in their "off" state, will have output voltage that is close to ground. (In an optional third phase, we can reduce the value of V_{DD} if such a change is required to match the load being driven; however, V_{DD} should be kept sufficiently high so that the hysteretic resistor at the selected line is not switched to "off".)

Appendix

We demonstrate here that conditions (i) and (ii) in Theorem 2.2 are tight in a worst-case sense.

We start by showing that generally, we cannot loosen condition (i). Suppose that at the gate at memory line ℓ we have $s_\ell + t_\ell > \vartheta$. Let $\boldsymbol{u}$ be the word in $\{0,1\}^n$ that was chosen by the designer to serve as the input word $\boldsymbol{a}$ which selects memory line ℓ; clearly, $\boldsymbol{u}$ covers $\boldsymbol{h}_\ell$. We show that the defects will in fact cause memory line ℓ to be deselected by that $\boldsymbol{u}$.

Let $\boldsymbol{u}_\ell = \boldsymbol{u} - \boldsymbol{e}'_\ell$ be the input word $\boldsymbol{a}$ $(= \boldsymbol{u})$ as seen by the defective gate at line ℓ (notice that defects of type `S-open`, if any, do not have any effect on $\boldsymbol{u}_\ell$). We have,

$$\langle \boldsymbol{h}_\ell, \overline{\boldsymbol{u}}_\ell \rangle = \langle \boldsymbol{h}_\ell, \overline{\boldsymbol{u}} + \boldsymbol{e}'_\ell \rangle = \underbrace{\langle \boldsymbol{h}_\ell, \overline{\boldsymbol{u}} \rangle}_{0} + \underbrace{\langle \boldsymbol{h}_\ell, \boldsymbol{e}'_\ell \rangle}_{\mathsf{w}(\boldsymbol{e}'_\ell)} = s_\ell > \vartheta - t_\ell \ ,$$

which means that memory line ℓ will be deselected.

Next, we turn to show the tightness of condition (ii), under the assumption that condition (i) holds. Let i and ℓ be such that

$$\langle \boldsymbol{h}_i, \overline{\boldsymbol{h}}_\ell \rangle = \mathsf{d}(\boldsymbol{h}_i, \boldsymbol{h}_\ell) = \mathsf{d}(\mathcal{C}) \ . \tag{A.1}$$

Suppose that the number of defects of type `S-open` at the gate at memory line i is such that

$$r_i = \mathsf{d}(\mathcal{C}) - \vartheta + t_i \ , \tag{A.2}$$

and that all defects of type `S-open` and `S-closed` at that gate occur at positions where ($\boldsymbol{h}_i$ is 1 and) $\boldsymbol{h}_\ell$ is 0: by (A.2) and condition (i) we have $r_i + s_i \le \mathsf{d}(\mathcal{C})$ and, so, (A.1) guarantees that we can find that many positions. We show that the defects will cause memory line i to be selected when we set the input word $\boldsymbol{a}$ to $\boldsymbol{h}_\ell$; this, in turn, will imply that line i will be selected by any word $\boldsymbol{u}$ chosen by the designer to select memory line ℓ, since such $\boldsymbol{u}$ must cover $\boldsymbol{h}_\ell$.

Let $\boldsymbol{a}_\ell = \boldsymbol{h}_\ell + \boldsymbol{e}_i$ be the input word $\boldsymbol{a}$ $(= \boldsymbol{h}_\ell)$ as seen by the defective gate at line i (defects of type `S-closed` have no effect on $\boldsymbol{a}_\ell$). Then,

$$\langle \boldsymbol{h}_i, \overline{\boldsymbol{a}}_\ell \rangle = \langle \boldsymbol{h}_i, \overline{\boldsymbol{h}}_\ell - \boldsymbol{e}_i \rangle = \underbrace{\langle \boldsymbol{h}_i, \overline{\boldsymbol{h}}_\ell \rangle}_{\mathsf{d}(\mathcal{C})} - \underbrace{\langle \boldsymbol{h}_i, \boldsymbol{e}_i \rangle}_{\mathsf{d}(\mathcal{C}) - \vartheta + t_i} = \vartheta - t_i \ ,$$

which means that memory line i will be selected.

In what follows, we demonstrate how to compute the probability that at least one of the conditions in Theorem 2.2 is not met, under the following probability model on the defects:

(i) Each input bit to a ϑ-threshold gate becomes defective of type `S-open` with probability p_{o}, and of type `S-closed` with probability p_{c}, independently of all other defects (at the same gate or at any other gate).

(ii) For the purpose of modeling defects of type `N-short`, we add to each threshold gate imaginary input bits, all set to 1, such that the total number of actual and imaginary input bits equals n. We then *mark* each of the imaginary input bit with probability p_{s}, independently of all other defects. A defect of type `N-short` at a gate changes an imaginary input bit into 0, but this can occur *only* at a marked bit[††] (thus, the number of marked bits at a gate is an *upper bound* on the number of defects of type `N-short` at that gate).

Let $\mathcal{C} = \{\boldsymbol{h}_0, \boldsymbol{h}_1, \ldots, \boldsymbol{h}_{M-1}\}$ be as in Theorem 2.2. The probability that the threshold gate at memory line i is subject to (exactly) r defects of type `S-open` and to s defects of type `S-closed` is given by substituting $w = \mathsf{w}(\boldsymbol{h}_i)$ in

$$\begin{aligned} P(r,s,w) &= \binom{w}{r,\, s} p_{\mathsf{o}}^r p_{\mathsf{c}}^s (1-p_{\mathsf{o}}-p_{\mathsf{c}})^{w-r-s} \\ &= \frac{w!}{r!s!(w-r-s)!} p_{\mathsf{o}}^r p_{\mathsf{c}}^s (1-p_{\mathsf{o}}-p_{\mathsf{c}})^{w-r-s} \; . \end{aligned}$$

The probability that t imaginary bits are marked at the same gate is given by substituting $m = n - \mathsf{w}(\boldsymbol{h}_i)$ in

$$Q(t,m) = \binom{m}{t} p_{\mathsf{s}}^t (1-p_{\mathsf{s}})^{m-t} \; .$$

The probability that the gate at memory line i satisfies conditions (i) and (ii') of Theorem 2.2 is obtained by substituting $\delta = \mathsf{d}(\mathcal{C})$ and $w = \mathsf{w}(\boldsymbol{h}_i)$ in

$$\pi(\delta, w, \vartheta) = \sum_{t=0}^{\vartheta} Q(t, n-w) \Big(\sum_{r=0}^{\delta-\vartheta-1} \sum_{s=0}^{\vartheta-t} P(r,s,w) \Big) \; ,$$

where we have used the number of marked imaginary bits as an upper bound on the number of defects of type `N-short` at the gate; when $p_{\mathsf{s}} = 0$ (i.e., when we do not expect defects of type `N-short`) the expression for $\pi(\cdot)$ simplifies to

$$\begin{aligned} \pi(\delta, w, \vartheta) &= \sum_{r=0}^{\delta-\vartheta-1} \sum_{s=0}^{\vartheta} P(r,s,w)) && \text{(A.3)} \\ &= \sum_{r=0}^{\delta-\vartheta-1} \sum_{s=0}^{\vartheta} \binom{w}{r,\, s} p_{\mathsf{o}}^r p_{\mathsf{c}}^s (1-p_{\mathsf{o}}-p_{\mathsf{c}})^{w-r-s} \; . && \text{(A.4)} \end{aligned}$$

[††] In particular, the change of imaginary bits into 0 can occur simultaneously in all the marked bits that are incident with the same address line—say, if that 0 results from the input value to that address line; under such circumstances, defects of type `N-short` will be statistically dependent, while the marked bits will still be independent.

(If the voltage margin is increased using the method described in Section 3.3, then the inner sum in (A.3)–(A.4) should be taken up to $\vartheta - t$, where t is as in (8).) Finally, the probability that at least one gate will not satisfy conditions (i)–(ii') is

$$\mathsf{P}_{\mathrm{fail}}(\mathcal{C}, \vartheta) \le 1 - \prod_{i=0}^{M-1} \pi(\mathsf{d}(\mathcal{C}), \mathsf{w}(\boldsymbol{h}_i), \vartheta) \; . \tag{A.5}$$

In particular, when $\mathcal{C}$ is an $(n, M, 2d, w)$-constant-weight code, we get

$$\mathsf{P}_{\mathrm{fail}}(\mathcal{C}, \vartheta) \le 1 - (\pi(d, w, \vartheta))^M \; .$$

Observe that $\mathsf{P}_{\mathrm{fail}}$ is the probability that either condition (i) or condition (ii') (or both) in Theorem 2.2 is not met. This probability is more conservative (i.e., larger) than the probability that the demultiplexer actually fails (in that it selects more than one memory line, or selects none). A slightly tighter formula (which is more tedious to compute) for the probability of failure is obtained by replacing each term $\pi(\mathsf{d}(\mathcal{C}), \mathsf{w}(\boldsymbol{h}_i), \vartheta)$ in (A.5) by

$$\pi^*(\mathcal{C}, i, \vartheta) = \sum_{t=0}^{\vartheta} Q(t, n - \mathsf{w}(\boldsymbol{h}_i)) P^*(\mathcal{C}, i, \vartheta - t) \; ,$$

where

$$P^*(\mathcal{C}, i, \mu) = \sum_{k=\mu+1}^{\mathsf{w}(\boldsymbol{h}_i)} \sum_{s=0}^{\mu} N(\mathcal{C}, i, k, \mu) \binom{k}{s} p_{\mathsf{o}}^{\mathsf{w}(\boldsymbol{h}_i)-k} p_{\mathsf{c}}^{s} (1 - p_{\mathsf{o}} - p_{\mathsf{c}})^{k-s} \; ,$$

and $N(\mathcal{C}, i, k, \mu)$ stands for the number of words $\boldsymbol{u} \in \{0,1\}^n$ that satisfy the following properties:

- $\boldsymbol{u} \subseteq \boldsymbol{h}_i$,
- $\mathsf{w}(\boldsymbol{u}) = k$, and—
- $\langle \boldsymbol{u}, \overline{\boldsymbol{h}}_\ell \rangle > \mu$, for all $\boldsymbol{h}_\ell \in \mathcal{C} \setminus \{\boldsymbol{h}_i\}$.

(Each such word $\boldsymbol{u}$ represents the positions in $\boldsymbol{h}_i$ that are unaffected by defects of type `S-open`.) For the case $p_{\mathsf{s}} = 0$ we get

$$\pi^*(\mathcal{C}, i, \vartheta) = P^*(\mathcal{C}, i, \vartheta) = \sum_{k=\vartheta+1}^{\mathsf{w}(\boldsymbol{h}_i)} \sum_{s=0}^{\vartheta} N(\mathcal{C}, i, k, \vartheta) \binom{k}{s} p_{\mathsf{o}}^{\mathsf{w}(\boldsymbol{h}_i)-k} p_{\mathsf{c}}^{s} (1 - p_{\mathsf{o}} - p_{\mathsf{c}})^{k-s}$$

(where the inner sum should be taken up to $\vartheta - t$ in case the voltage margin is increased using the method in Section 3.3).

Acknowledgment

The authors wish to thank Joe Straznicky for providing us with helpful insights regarding the implementation of comparators as discussed in Section 3.

References

[1] E. Agrell, A. Vardy, K. Zeger, *Upper bounds for constant-weight codes, IEEE Trans. Inform. Theory,* 46 (2000), 2373–2395.

[2] V. Beiu, J.M. Quintana, M.J. Avedillo, *VLSI implementations of threshold logic—a comprehensive survey, IEEE Trans. Neural Networks,* 5 (2003), 1217–1243.

[3] M. Blaum, H.C.A. van Tilborg, *On t-error-correcting/all unidirectional error detecting codes, IEEE Trans. Comput.,* 38 (1989), 1493–1501.

[4] F.J.H. Böinck, H.C.A. van Tilborg, *Constructions and bounds for systematic tEC/AUED codes, IEEE Trans. Inform. Theory,* 36 (1990), 1381–1390.

[5] B. Bose, D.K. Pradhan, *Optimal unidirectional error detecting/correcting codes, IEEE Trans. Comput.,* 31 (1982), 564–568.

[6] A.E. Brouwer, J.S. Shearer, N.J.A. Sloane, W.D. Smith, *A new table of constant weight codes, IEEE Trans. Inform. Theory,* 36 (1990), 1334–1380.

[7] J. Bruck, M. Blaum, *New techniques for constructing EC/AUED codes, IEEE Trans. Comput.,* 41 (1992), 1318–1324.

[8] J. Bruck, M. Blaum, *Coding for skew-tolerant parallel asynchronous communications, IEEE Trans. Inform. Theory,* 39 (1993), 379–388.

[9] J. Bruck, M. Blaum, L.H. Khachatrian, *Constructions of skew-tolerant and skew-detecting codes, IEEE Trans. Inform. Theory,* 39 (1993), 1751–1757.

[10] A. DeHon, H. Naeimi, *Seven strategies for tolerating highly defective fabrication, IEEE Design & Test Comput.,* 22 (2005), 306–315.

[11] H. Heil, J.-M. Gantioler, R. Sander, *Integrated comparator circuit with four MOSFETS of defined transfer characteristics*, US Patent 6,057,712, 2000.

[12] J.N. Hogan, R.M. Roth, *Fault-tolerant neighbor-disjoint addressing scheme,* HP Laboratories Tech. Report No. HPL-2001-88(R.1), 2001.

[13] J.N. Hogan, R.M. Roth, *Fault-tolerant address logic for solid state memory,* US Patent 6,459,648, 2002.

[14] P.J. Kuekes, W. Robinett, R.M. Roth, G. Seroussi, G.S. Snider, R.S. Williams, *Resistor-logic demultiplexers for nanoelectronics based on constant-weight codes, Nanotechnology,* 17 (2006), 1052–1061.

[15] P.J. Kuekes, W. Robinett, G. Seroussi, R.S. Williams, *Defect-tolerant interconnect to nanoelectronic circuits: internally redundant demultiplexers based on error-correcting codes, Nanotechnology,* 16 (2005), 869–882.

[16] P.J. Kuekes, W. Robinett, G. Seroussi, R.S. Williams, *Defect-tolerant demultiplexers for nano-electronics constructed from error-correcting codes, Appl. Phys. A,* 80 (2005), 1161–1164.

[17] P.J. Kuekes, W. Robinett, R.S. Williams, *Improved voltage margins using linear error-correcting codes in resistor-logic demultiplexers for nanoelectronics, Nanotechnology,* 16 (2005), 1419–1432.

[18] P.J. Kuekes, W. Robinett, R.S. Williams, *Defect tolerance in resistor-logic demultiplexers for nanoelectronics, Nanotechnology,* 17 (2006), 2466–2474.

[19] D. Lubell, *A short proof of Sperner's theorem, J. Combin. Theory,* 1 (1966), 299.

[20] F.J. MacWilliams, N.J.A. Sloane, *The Theory of Error-Correcting Codes.* Amsterdam: North-Holland, 1977.

[21] H. Naeimi, A. DeHon, *A greedy algorithm for tolerating defective crosspoints in nanoPLA design, Proc. Int'l Conf. on Field-Programmable Technology (ICFPT'2004),* Brisbane, Australia (2004), 49–56.

[22] H. Naeimi, A. DeHon, *Fault tolerant nano-memory with fault secure encoder and decoder, 2nd Int'l Conf. on Nano-Networks (Nano-Net'2007),* Catania, Italy (2007).

[23] D.K. Pradhan, *A new class of error-correcting/detecting codes for fault-tolerant computer applications, IEEE Trans. Comput.,* 29 (1980), 471–481.

[24] W. Robinett, P.J. Kuekes, R.S. Williams, *Defect tolerance based on coding and series replication in transistor-logic demultiplexer circuits, IEEE Trans. Circuits Syst. I,* 54 (2007), 2410–2421.

[25] G.S. Snider, *Computing with hysteretic resistor crossbars, Appl. Phys. A,* 80 (2005), 1165–1172.

[26] G.S. Snider, P.J. Kuekes, *Nano state machines using hysteretic resistors and diode crossbars, IEEE Trans. Nanotechnology,* 5 (2006), 129–137.

[27] E. Sperner, *Ein Satz über Untermengen einer eindlichen Menge, Math. Zeitschrift,* 27 (1928), 544–548.

[28] A. Tangel, K. Choi, *"The CMOS inverter" as a comparator in ADC designs, Analog Integr. Circuits Signal Process.,* 29 (2004), 147–155.

[29] T.R. Viswanathan, *Fast comparator circuit,* US Patent 5,532,628, 1996.