Robust Nanowire Decoding

Eric Rachlin

Abstract

In recent years, a number of nanoscale devices have been demonstrated that act as wires and gates. In theory, these devices can interconnect to form general purpose architectures. Unfortunately, our ability to place individual devices is poor, and device reliability may be substantially lower than that of current CMOS technology. Nanoscale architectures must be designed with these limitations in mind. Designs must account for extremely high levels of variation in both device placement and operation.

It remains uncertain exactly what these robust nanoscale architectures will look like. At a minimum, however, they will require control over individual nanoscale wires. A device which controls a set of parallel nanowires with a second set of input wires is called a nanowire decoder. In order to interface nanotechnology with existing lithographically produced technology, the decoder's input wires can be large, mesoscale wires.

In this paper we provide a general model for nanowire decoders. We use this model to give the conditions that the decoders must meet. We also describe how to create fault-tolerant decoders for use in both memories and circuits. Finally we discuss the overhead and testing procedures required to ensure that stochastically assembled decoders behave deterministically.

The problems we address are very relevant to current research in nanoscale computing. They also highlight the more general issues of stochastic assembly and fault-tolerance. Both issues will likely remain a fundamental characteristic nanoscale computing, and as such become a primary focus of the budding field of computational nanotechnology.

1 Introduction

One nanometer, or 10^{-9} meters, is the length of a single sugar molecule, and a cubic nanometer provides only enough room for a few hundred carbon atoms. It may never be possible to create novel arrangements of subatomic particles, and as such, a nanometer represents the approximate lower limit on the size of technology.

Nanometer-scale technology, or "nanotechnology", has a wealth of applications and nanoscale computing is among the most prominent. The dream of nanoscale computing was first articulated by Richard Feynman in his 1959 speech given to the American Physical Society. He argued that no known physical law would prevent the room-sized computers of the 50's from being replaced with far more powerful, pin-sized computers built from nanoscale components. As many at the time realized, general purpose computers would become far more interesting when their computing power increased by several orders of magnitude. This

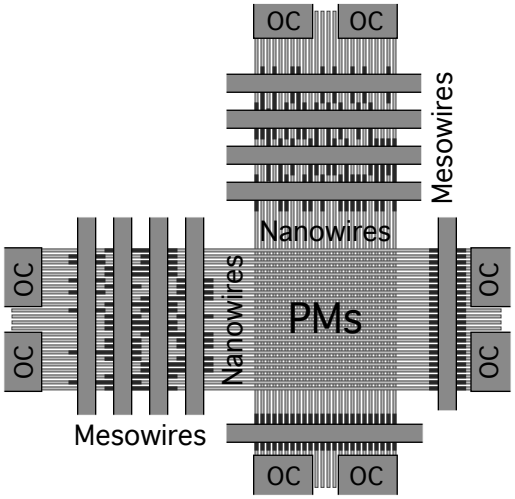

Figure 1. A crossbar with two parallel sets of NWs controlled by mesoscale address wires. FETs (or possibly diodes) are defined at the intersection of lightly doped (dark) NW regions with mesowires. Ohmic contacts (OCs) are made at ends of each set of NWs. Data is stored in the conductivity of molecular switches at crosspoints, intersections of orthogonal NWs.

has long since proven true, but we are only now approaching the nanoscale devices that Feynman asserted we could produce.

Producing architectures from nanoscale devices is not simply a matter of substituting tiny wires and gates in today's architectures. Designers of nanoscale architectures must find ways to interconnect millions, or billions of devices, even when our ability to place each individual devices is poor. Furthermore, as we continue to push the limits of what can be reliably manufactured, we must find new ways to mitigate device variation. If nanoscale architectures are to be realized any time soon, they will have to function correctly even when individual devices fail.

These are some of the fundamental challenges that face the new field of computational nanotechnology. They are not specific to a particular type of manufacturing process or even to silicon. The analysis in this paper specifically addresses today's most viable nanotechnology, the silicon nanowire (NW). Nonetheless, it highlights some much more general issues which suggest very promising areas for future research.

1.1 Nanoscale Architectures

Although no nanoscale architectures have been produced, many believe that they will incorporate stochastic assembly, reconfigurablity and fault-tolerance [9, 13]. Rather than speculate further as to how these trends will manifest themselves, we ground our work in a single, particularly viable nanotechnology, the **nanowire crossbar** (See Figure 1). To date, the crossbar is the only nanoscale architectural component to be produced [29, 17, 3]. Multiple crossbar-based architectures have been considered, demonstrating how crossbars can serve as a basis for both memories and circuits [7, 16, 10, 12].

A NW crossbar consists of two orthogonal sets of parallel NWs, separated by a layer of molecular devices. Each pair of perpendicular NWs provides control over the molecules at their crosspoint. To turn on an

individual pair of orthogonal NWs, a voltage is applied to a group of consecutive NWs along each dimension of the crossbar. Sets of address wires (AWs) are then used to turn off most of the NWs within each group.

Operation of the crossbar is described in detail at the end of Section 2. For now, we merely highlight the fact that the a crossbar's functionality hinges on its ability to control individual NWs along each dimension. Controlling individual NWs is a basic requirement of any nanoscale architecture that exploits the high density nanotechnology provides. In the near-term, we also expect nanoscale architectures to be highly regular. Like a crossbar, they will likely consist of a large number of parallel NWs with devices at their crosspoints.

For these reasons, the NW crossbar serves as an excellent concrete example of the control nanoscale architectures require over NWs. The device which allows a set of AW inputs to select one NW among a group of parallel NWs is called a **nanowire decoder**. In this paper we explore the requirements a NW decoder must meet. We also explain how these requirements can be met, even when decoder assembly is stochastic and nanoscale connections fail.

1.2 Overview of Content

Section 2 provides an overview of current NW technology. It describes what types of NWs can currently be produced, and how these NWs can be controlled. As explained, these control technologies can be used to form multiple types of NW decoders. In order to analyze and compare these decoders, we present a general decoder model in Section 3. In our model, we describe three ways in which AWs control NWs:

- Case A: Each AW provides complete control over some NWs.
- Case B: Each AW increases a NW's resistance by some amount.
- Case C: Each AW provides total control over some NWs, except when errors occur, in which case its effect on some NWs is unpredictable.

In Section 4 we introduce binary codewords and analyze case A. In Section 5 we present two families of binary codes that are relevant to NW decoders. In Section 6 we introduce real-valued codewords to analyze case B, then compare our results to case A. Finally in Section 7 we introduce errors to binary codewords and analyze case C. The result is a practical model that is easier to work with than case B, but more realistic than case A. In our analysis, we introduce the notion of "balanced Hamming distance", which accurately describes how NWs remain controllable even in the presence of errors.

In Section 8 we relate NW codewords to stochastic decoders. We present bounds on the amount of redundancy required to generate enough valid codewords with high probability. We also highlight the tradeoff between codespace size, NW redundancy and the amount of control circuitry required. Section 9 explains how stochastically generated codewords can be discovered through testing. Finally Section 10 summarizes our results and highlights the general issues that we feel are key concerns for computational nanotechnology.

2 Technology overview

In order to justify the applicability and generality of the decoder model presented in Section 3, we first review a variety of relevant technologies. We begin with a review of NW production, then discuss approaches for controlling NWs, and finally describe how these control technologies can create multiple types of NW decoder. We conclude the section with a description of how a NW decoders functions in the context of a crossbar.

2.1 Nanowires

In this subsection we review current methods of NW production. We classify the resultant NWs into two categories; undifferentiated and differentiated. Undifferentiated NWs are typically produced directly on a chip, and are all identical. In contrast, differentiated NWs are grown off chip, then deposited. By growing different sets of NWs separately, each set can be given different electrical characteristics, which facilitates the production of a decoder. Methods for controlling both differentiated and undifferentiated NWs are described in Section 2.2.

2.1.1 Undifferentiated Nanowires

A significant amount of research has focused on a producing large numbers of identical silicon NWs directly on a chip. The resulting **undifferentiated nanowires**, may be lightly or heavily doped, but must all be indentical. There are several ways of producing undifferentiated NWs.

SNAP: A particularly successful approach for growing undifferentiated NWs is the SNAP method [21]. Here a superlattice (multiple crystalline blocks of material) is formed consisting of thin alternating layers of two materials, such as Aluminum Galium Arsenide and Galium Arsenide. One type of material is etched back to create nanoscale notches. Metal is then deposited onto these notches, and the superlattice is pressed onto a silicon chip that contains a thin layer of adhesive. The metal sticks to the adhesive, creating a pattern of nanoscale metal wires that can be treated as in photolithography.

Alternatively, the superlattice (without metal) can be pressed on a soft polymer, creating a pattern of hills and valleys. When a thin layer of polymer is removed, only the hills remain, creating a pattern very similar to that of the metal wires just described. In either case, long straight silicon NWs can be produced on the chip.

Porous Membranes: A second approach to producing undifferentiated NWs which is unconstrained by photolithography involves porous alumina membranes. These membranes contain a regular hexagonal pattern of nanoscale pores. Within each pore, a NW can be grown [4]. If the porous membrane is turned on its side, uniform parallel NWs can be placed on the surface of a chip. Unlike SNAP, the resulting NWs are single crystal, and thus may have improved conductivity.

On Chip Catalysts: Undifferentiated NWs have also been grown on chip from a pattern of seed catalysts (such as gold clusters) with varying degrees of success. In some cases, straight, uniform NWs have been grown upward, perpendicular to the surface of the chip [31]. Even so, it is unclear how these wires can be incorporated in to a nanoscale architecture.

2.1.2 Differentiated Nanowires

In contrast to undifferentiated NWs **differentiated nanowires** are grown separately off chip, then deposited fluidically [29]. If many sets of NWs are grown, then collected in a large ensemble, all NWs in the ensemble need not be identical. NWs of different types can be produced though variations in both axial and radial growth.

Axially Encoded Nanowires: A pattern of doped regions can be placed along the axis of a silicon NW as it grows from a seed catalyst through a vapor-liquid-solid (VLS) process [15, 32, 2]. In [15], silane molecules fall onto gold clusters, causing a silicon NW to grow axially. As the NW grows, the amount of dopant in the gaseous mixture is modulated. The resultant sequence of lightly and heavily doped regions can be precisely controlled, producing an axial encoded NW. In Section 2.3 we describe how these axial encodings can form the basis for a NW decoder.

Radially Encoded Nanowires: As with axially encoded NWs, radially encoded NWs are produced though a VLS process. Instead of axial doping patterns, sequences of shell materials differentiate the NWs. First a set of lightly doped silicon NWs are grown, then a shell material is deposited around each NW [20]. As the NW grows radially, the shell materials can vary. As shown in [27] and described below, shells made from independently etchable materials create a NW decoder when exposed to certain sequences of etchants.

It is also possible to produce NWs that are both axially and radially encoded. These "hybrid encoded NWs" are considered in [27], but do not yield a clear advantage over either approach when used in isolation.

2.2 Nanowire Control

Given our ability to produce a large number of parallel NWs, we now consider ways of controlling the current across those NWs. We have two options; use direct contact to apply a voltage across NWs, or use a field to control the resistance of NWs. Here we describe how each form of control can be achieved.

2.2.1 Voltage Control Via Contacts

There are several proposed approaches for controlling the voltage across NWs. The most basic is to apply a potential across an entire set of parallel NWs. The following approaches provide more fine-grained control.

Ohmic Contacts: A pair of lithographically produced ohmic contacts can be produced at both ends of a set of parallel NWs. This allows a potential to be applied by across all NWs. Even when NW pitch is small, we expect ohmic contacts to provide voltage control over groups of about 10 to 20 NWs. Furthermore, since sets of contacts are produced lithographically, they can be reliably controlled using standard CMOS circuitry.

CMOL: Unlike large lithographically produced contacts, nanoscale contacts can apply a voltage to a single NW. A scheme known as CMOL strives to do just this [28]. In CMOL, a grid of lithographically produced AWs controls nanoscale "pins" at AW crosspoints. These pins make contact with an overlying layer of NWs. In theory, a properly oriented grid of AWs would have at least one pin in contact with each nanowire. Unfortunately, this scheme has not been demonstrated and seems to require extremely precise alignment between NWs and pins. The alignment problem becomes even more daunting when we wish to control multiple layers of NWs (as in a crossbar).

Diode Contacts: Fine-grained control over NW potentials can also be achieved if NWs are connected to AWs with diodes. Here each AW is connected to a subset of the NWs and can apply a potential to just those NWs. Multiple AWs can be used simultaneously to apply a potential to any NW connected to at least one of the AWs. Multiple AWs, when used in conjunction with ohmic contacts, can generate a current current along the entire length of an individual NW (see Section 2.3.5). This form of control is particularly viable when NWs are used as AWs, since diode connections may be programmed like bits stored in a crossbar memory (see Section 2.4). Nanoscale AWs, however, would themselves require a NW decoder to control.

2.2.2 Resistive Control Via Field Effect

When a pair of ohmic contacts applies a potential across a group of consecutive NWs, each NW in the group carries an equal current unless their resistances vary. The resistance of a lightly doped silicon NW significantly increases in the presence of a sufficiently strong electric field. If a lithographically produced AW is laid down across a set of lightly doped NWs, the NWs only conduct when the AW is not producing an electric field. In this way, the AW forms a field-effect transistor (FET) with each NW.

If each AW forms an FET with only some of the NWs, multiple AWs can be used simultaneously to gain fine-grained control over NW resistances. For example, some NWs may contain lightly doped regions under some AWs and lightly doped regions under others. If a subset of the AWs all produce an electric field simultaneously, all NWs with a lightly doped region under any one of those AWs will have a high resistance. Similar behavior results if NWs are lightly doped along their entire length, but portions of each NW are shielded from the fields of some AWs.

2.3 Proposed Decoders

Different types of NWs can be combined with different forms of control to produce a number of NW decoders. In each decoder, a voltage is applied across a number of NWs, then address wires (or some other means of control) prevent some of those NWs from carrying a current. In this way, the decoders allow us to address one (or a few) NWs.

2.3.1 Axial and Radial NW Decoders

Axially and radially encoded NWs enable a NW to form FETs with some NWs but not others. First consider an **axial decoder** in which a set of axially encoded NWs are deposited in parallel on a chip. A decoder is created by placing ohmic contacts at either end of the NWs, and AWs perpendicularly across the NWs (see Figure 1) [11].

When NWs are deposited, they are randomly selected from an ensemble with many copies of each axial encoding. If two NWs have different axial encodings, they will have lightly doped regions under different AWs. Even if two NWs have the same encoding, their doped regions may have shifted relative to each other during fluidic deposition. In either case, properly chosen encodings will allow different subsets of AWs to make the different NWs nonconducting. When a voltage is applied across all NWs using the ohmic contacts, only those NWs that have not been made nonconducting are addressed by the decoder.

In an axial decoder, there is no guarantee that one of a NW's lightly doped region won't lie partially under a AW. Such misalignment can prevent AWs from providing good control over that NW. Radially encoded NWs allow for decoders that do not suffer from this shortcoming. In a **radial decoder**, radially encoded NWs are deposited on a chip, but before each AW is laid down, a specific sequence of etchants is applied to the region under each AW. Etching sequences are chosen to remove all of the shells around some NWs but not others. Under each AW, the lightly doped cores of only some NWs are exposed, allowing the AW to control only those NWs. If the shell materials and etching sequences are chosen properly, the radial decoder will be able to simulate any axial decoder, but without the possibility of misalignment. A detailed comparison of axial and radial decoders is given in [27].

2.3.2 Mask-Based Decoders

Radial and axial decoders require that NWs be differentiated. To control undifferentiated NWs, such as those produced by SNAP, a **mask-based decoder** has been proposed [1, 26]. In a mask-based decoder, lightly doped undifferentiated NWs are placed between ohmic contacts, and controlled with perpendicular AWs. To prevent each AW from forming an FET with every NW, rectangular regions of high-K and low-K dielectric are placed between NWs and AWs using a mask. When an AW generates an electric field, the field is amplified by the regions of high-K dielectric under the AW. Only NWs under those regions are controlled by that AW.

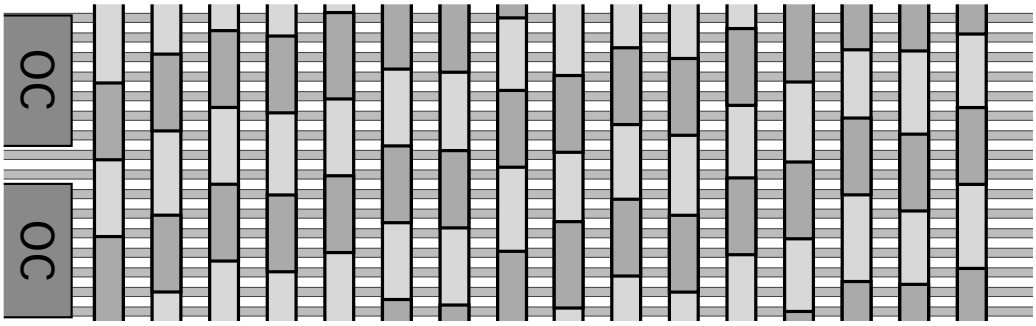

Figure 2. A mask-based decoder in which dark gray rectangles indicate regions of high-K dielectric underneath address wires. Regions are produced from a mask containing a cyclic pattern of rectangular holes, but random variation in lithographic manufacturing causes regions to shift from their intended location. This necessitates redundancy to gain control over all NWs.

In a mask-based decoder, dielectric regions are much larger than the thickness of NWs. Also, their position cannot be controlled with nanoscale precision. Although the mask used to produce the regions provides some control over where regions land, redundancy is required to ensure that many NWs are controllable with high probability. In general, multiple copies of each region are produced under successive AWs, each shifted by a small, but random amount (see Figure 2). As with axial decoders, misalignment is also a concern. If the endpoint of a high-K dielectric region lies in the middle of a NW, the AW that lies on top of that region may provide only partial control over that NW.

2.3.3 Random Contact Decoders

Although the decoders discussed so far are assembled stochastically, they still provide some control over which AWs control which NWs. In axial and radial decoders, we are free to choose some NW encodings over others. In a mask-based decoder, the mask allows us to aim each high-K dielectric region over some subset of the NWs. By contrast, a **random contact decoder** treats each NW/AW junction as an independent random variable. Each AW controls each NW with some fixed probability, p.

There are multiple ways to produce random contacts between AWs and NWs. One proposed approach is to randomly deposit impurities (such as gold particles) onto undifferentiated NWs [30]. Another approach is to randomly deposit small regions of high-K dielectric, or alternatively, randomly etch holes in a low-K dielectric. Also, a random contact decoder can be made from axially encoded NWs. If many sets of axially encoded NWs are produced with randomly placed lightly doped regions, each NW/AW junction can be treated as an independent random variable. As a result, analysis of random-contact decoders provides bounds that apply to axial (and hence radial) decoders as well.

2.3.4 MNAB

A final somewhat different approach for controlling sets of undifferentiated NWs is to exploit differences in their spacial location. The resistance of a lightly doped undifferentiated silicon NW is substantially

increased in the presence of a large electric field. Suppose two mesoscale electrodes are placed to the left and right of a small group of N parallel NWs. If each electrode produces a field simultaneously, the fields are summed. The net field will has minimum value at some location between the two electrodes.

If the shape and magnitude of the field produced by the two electrodes can be carefully controlled, the net field can have a minimum value located at any one of the N NWs. The shape of the field can be such that only that one NW remains conducting. All other NWs, which experience a stronger electric field, experience an increase in resistance.

This type of decoder, termed **MNAB** [14], is promising. It may, however, be very sensitive to small variations in NW location and in the amount of dopant in each undifferentiated NW. When implemented on chip, each pair of electrodes cannot be tuned to produce an arbitrary set of electric fields. Instead, each electrode would produce one of a small number of predetermined fields. This may limit the number of NWs each pair of electrodes can reliably control.

2.3.5 Programmable NW Decoders

All of the above decoders (with the possible exception of MNAB) are assembled stochastically. It is not known in advance which sets of AWs will address a NW. Stochastic assembly is a natural consequence of our poor ability to place nanoscale features. Although we expect this to remain true for some time, it may be possible to use a stochastically assembled decoder to gain the fine-grained control required to deterministically program a second decoder. There have been several proposals for doing this.

In one proposed approach, an axial decoder is formed along one section of the NWs, while a second section contains floating-gate devices between NWs and a set of auxiliary AWs [8]. The stochastically assembled axial decoder allows individual NWs to conduct, which in turn permits them to be coupled with subsets of the auxiliary AWs using the floating-gate devices. To program the second decoder, NWs would be addressed while subsets of the auxiliary AWs carry a large voltage. The floating-gate devices couple these AWs to the NW, effectively programming FETs at specified locations along the NW.

A second approach to programming a decoder uses hysteretic molecules to couple NWs to other NWs [18]. Just as diode connections can be programmed at the crosspoints of a crossbar-based memory, they can connect NWs to nanoscale AWs. In both cases, however, both sets of NWs must be controllable by a decoder.

In a **diode-based decoder**, AWs do not increase the resistance of NWs. Instead, current flows off of NWs onto whichever AWs are grounded. As with FET-based NWs, a pair of ohmic contacts applies a voltage across a set of NWs. If the ohmic contact placed near the AWs is placed at V_{vdd}, and the other contact is placed at V_{gnd}, current flows along the entire length of all NWs.

AWs and diode contacts are assumed to have a substantially smaller resistance than that of the NWs. If a single AW is grounded, almost all current in each NW it is connected to will flow into that AW (here diode connections cause current to flow from NWs to AWs). If a subset of the AWs are grounded, only NWs not connected to any of the AWs continue to carry a large current along their length.

Returning to FET-based decoders, a one-time programmable decoder can also be created using radially encoded NWs. A standard radial decoder reveals which radial encodings are present on the NWs which have been randomly deposited on the chip. Once these shell sequences are known, a second radial decoder could be produced using etching sequences specifically tailored to those shell sequences which are present. [24].

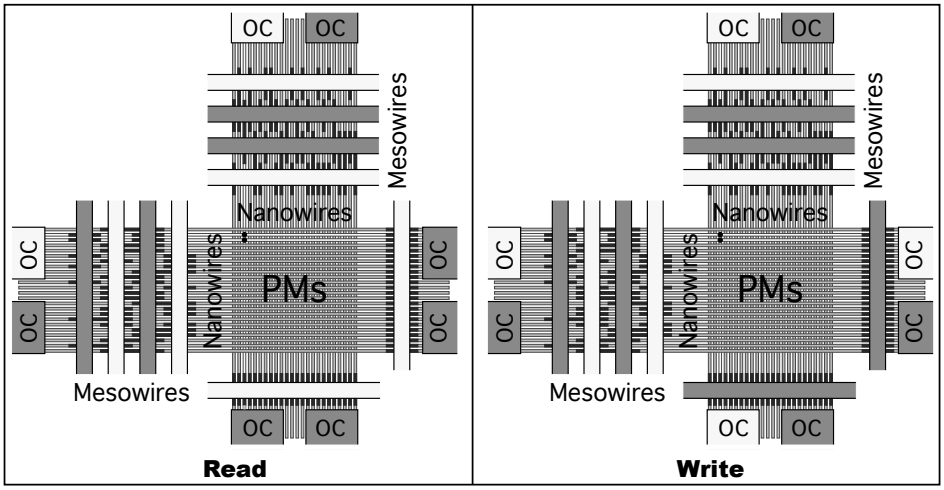

Figure 3. A crossbar-based memory in which ohmic contacts (OCs) and mesoscale address wires (mesowires) read and write data to programmable molecules located at nanowire crosspoints. In a read operation, current flows through the crosspoints of whichever NWs carry a current. The amount of current reveals the value stored at the crosspoints. In a write operation, NWs along each dimension apply a larger electric field across their crosspoints. The direction of the field determines the value stored at the crosspoints.

2.4 Operation of a Crossbar

As illustrated in Figures 1 and 3 a crossbar consists of two layers of parallel NWs separated by a layer of molecular switches [5, 6]. These devices act as programmable diodes that can switch between states of high and low resistance. If a pair of perpendicular NWs applies a small voltage across their crosspoint, the resistance of the crosspoint can be measured. If instead, a larger voltage is applied, the resistance can be set.

2.4.1 Write Operations

In a crossbar write operation, the resistance of a crosspoint is set to either a high or low value. To apply a large potential across the crosspoint of two NWs, a charge is placed on each wire using a NW decoder. One ohmic contact of each decoder is connected to ground, while the other applies a large voltage. Only those NWs which the decoders address accumulate a significant charge. The result is a large potential across their crosspoints, the direction of which determines the state of the crosspoints. Notice that if the NW decoders address multiple NWs at once, the same value can be written to multiple crosspoints simultaneously.

2.4.2 Read Operations

In a crossbar read operation, a smaller voltage is applied across the NWs to measure the resistance of crosspoints. A read operation is identical to a write operation, except that NWs along each connection are disconnected from ground. This ensures that current flows through the crosspoint. To perform this disconnect, a single AW, which forms an FET with all NWs, is placed at the other end of the crossbar.

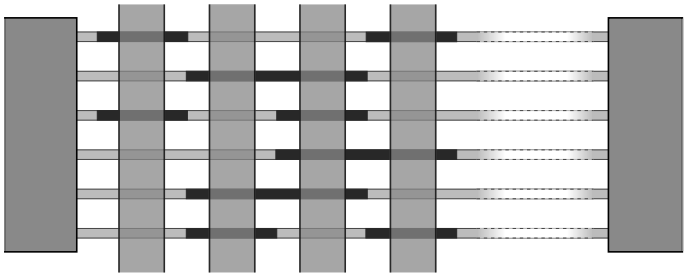

Figure 4. A simple nanowire decoder in which a pair of ohmic contacts apply a voltage across a set of parallel NWs. Perpendicular AWs each provide control over a subset of the nanowires.

Multiple crossbar read operations can also implement logic. If multiple wires along one dimension carry a current, the parallel read acts as a WIRED-OR. Current will flow if any of the crosspoints is set to a conducting state. This operation, when combined with negation (provided by FETs formed using lightly doped regions) provides the basis for a programmable logic array [7].

3 The Decoder Model

Given the wealth of decoding technology under consideration, we propose the following general definition of a NW Decoder. Our initial definition, given below, is broad enough to apply to all technologies under consideration. It is necessarily vague, however, concerning how AWs control NWs. Several forms of AW/NW control are described later in this section.

3.1 The Simple Nanowire Decoder

Definition 3.1 *In a* **simple nanowire decoder** *(see Figure 4):*

1. *M AWs control N NWs. The NWs are tightly packed and (at least partially) aligned, but are not in electrical contact with one another.*
2. *A voltage can be applied along the length of all NWs simultaneously. In the absence of AW control, all NWs conduct, effectively behaving like a single wire.*
3. *Each AW provides control over some subset of the NWs. An AW* **controls** *a NW if the AW can substantially reduce the amount of current flowing from one end of the NW to the other. When an AW reduces the current flowing through those NWs it controls, it is said to be* **activated***. When an AW is not activated, it has no effect on any NWs.*
4. *When multiple AWs are activated, each NW which is controlled by any one of the AWs experiences a reduction in current. The subset of NWs which do not experience a reduction in current are said to be addressed by the decoder.*

In general, we expect the N NWs to be laid down in parallel, and the M AWs to lie perpendicularly across the NWs. The particular geometry of the decoder is unimportant, however, as long as the behavior of

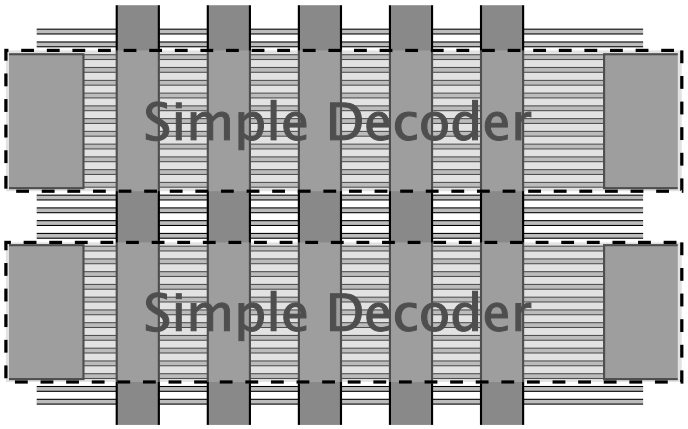

Figure 5. A composite nanowire decoder in which multiple simple decoders are placed in parallel, allowing them to share a common set of inputs.

the decoder fits with our general model. In MNAB, for example, control is not provided by a perpendicular set of AWs, but rather mesoscale electrodes placed on either side of the NWs. This is perfectly acceptable.

Notice that AW control is "subtractive". If no AWs are activated, all N NWs carry a current. As AWs are turned on, subsets of NWs are turned off. If each AW provides control over approximately half the NWs, M can be logarithmic in N (see Section 5). By contrast, an "additive" decoder, in which each AW made a set of NWs carry a current, would require at least one AW per NW to be individually controlled. CMOL can be thought of as an additive decoder. In the context of our subtractive model, however, CMOL can also be thought of as a compound decoder (see below) for which $N = 1$.

If each NW can be controlled by a single AW, decoder operation is straightforward. Any NW (or subset of NWs) can be turned off by activating the corresponding AW (or subset of AWs). Unfortunately, as was noted in Section 2 in our discussion of CMOL, one-to-one contacts are difficult to produce. They appear to require a level of manufacturing precision that is not yet available. Since this is primarily an engineering problem, we do not focus on CMOL in this paper.

3.2 The Composite Nanowire Decoder

At present, simple decoders are manufactured stochastically. Each AW provides control over a subset of NWs, but these subsets are chosen randomly according to some distribution. As the number of NWs increases, the amount of variability in the manufacturing process increases, as does the required number of AWs. For both these reasons, it is useful to combine multiple simple NW decoders to form a single composite NW decoder.

Definition 3.2 *A* **composite nanowire decoder** *consists of g simple NW decoders, arranged in parallel (see Figure 5). Each simple decoder can be turned on or off independently. Each simple decoder, when on, applies a voltage across a different set of N NWs. All simple decoders share a single set of M AWs.*

In a composite NW decoder, the AWs only address NWs controlled by the simple decoders that are on. Since we are free turn on one simple decoder at a time, a composite NW decoder is at least as powerful as each simple decoder used in isolation. The composite NW decoder, however, provides a substantial savings in space. Using the same set of AWs across many simple decoders allows us to consider designs for which $M > N$, even if each AW is substantially larger than each NW. A composite NW decoder is practical as long as the M AWs do not take up significantly more space than all gN NWs.

3.3 Models of Control

Thus far, our decoder definitions provide a qualitative summary of many viable decoding technologies. Before using these definitions to derive quantitative results, we must replace the vague notion of "control" with mathematically rigorous description of how AWs affect NWs. The following subsections present the three control models we consider. The first model is idealized, and as a result, simple to work with. The second model is more realistic, but cumbersome. The third and final model is a compromise between the two. It is more practical and realistic than the first, but more appealing to work with than the second.

3.3.1 Ideal Model

In the **ideal model** of NW control, an AW either completely turns off a NW, or has no effect on that NW. When an AW is activated, it completely prevents some subset of the NWs from carrying a current. It has no effect on the remaining NWs. This model is investigated in the Section 4

The ideal model does a good job of modeling diode-based decoders. In these decoders an AW which is activated drops the voltage across all NWs it controls to near zero. The ideal model also does a good job of modeling FET-based decoders when the FETs have a sufficiently large on/off ratio. In this case, NWs controlled by activated AWs are effectively nonconducting. A proof of the required on/off ration in given in Section 6.4.

3.3.2 Resistive Model

Since FETs play a prominent role in multiple decoding technologies, we also consider a more realistic **resistive model**. Here each AW which is activated increases each NWs resistance by some amount. If the AW does not control a NW, the increase will be very small (possibly 0). If the AW provides good control over a NW, the increase will be high. The total resistance of a NW is given by its initial resistance plus the increase associated with each activated AW. This model is explored in Section 6.

3.3.3 Ideal Model with Errors

The ideal model is easy to work with and provides clean results. The resistive model, however, is more accurate. In order to make the ideal model physically realistic, we introduce the notion of errors. In the ideal model with errors, some AWs provide only partial control over some NWs. Errors allow us to model imperfections in decoder manufacture. They also allow us to relate the resistive model to the ideal model. This model is the focus of Section 7. It provides insight into how fault-tolerant nanoscale architectures may one day be designed.

4 Ideal Decoders

If a NW decoder obeys the ideal model of control, each AW provides total control over a subset of NWs, and no control over the remaining NWs. We call this an **ideal nanowire decoder**. In this section we employ "binary codewords" to describe these decoders and the requirements they must meet.

4.1 Binary Codewords

Consider a simple ideal NW decoder with M AW inputs and N NW outputs.

Definition 4.1 *A* **binary codeword** *is an M-bit vector associated with each NW. The j^{th} bit of a NW's codeword is 1 if and only if that NW is controlled by the j^{th} AW, and 0 otherwise.*

Let $\boldsymbol{c^i}$ be the codeword associated with NW $\boldsymbol{n_i}$, $1 \leq i \leq N$, and let c^i_j, $1 \leq j \leq M$, denote the j^{th} bit of that codeword. For each codeword, $c^i_j = 1$ (0) if and only if NW $\boldsymbol{n_i}$ is controlled (unaffected) by the j^{th} AW.

Let $\boldsymbol{a}$ be the M-bit input supplied to the decoder. In other words, $a_j = 1$ if and only if AW a_j carries a voltage. We refer to $\boldsymbol{a}$ as an **activation pattern**. An activation pattern turns off NW $\boldsymbol{n_i}$ if and only if there exists a j such that $a_j = 1$ and $c^i_j = 1$. Equivalently, $\boldsymbol{a}$ turns off NW $\boldsymbol{n_i}$ if and only if $\boldsymbol{a} \cdot \boldsymbol{c^i} = \sum_{j=1}^{M} a_i c^i_j \geq 1$, where addition is over the integers.

Notice that inputs, as well as codewords, are binary. Each AW is in one of two states. This assumption fits well with most of the decoding technologies we have described. The one notable exception is MNAB, in which each electrode can produce electric fields of multiple strengths. In this way, each electrode can be in one of a small number of states. One way to model MNAB with our binary model, we would represent each pair of electrodes as multiple AWs, one for each electric field the electrodes can colletively produce. We would also add the constraint that only one of these AWs can be activated at a time.

Before continuing, we also note that decoding technologies assign codewords to NWs stochastically. Different technologies provide varying degrees of control regarding what codewords are assigned to NWs. This is discussed in detail in Section 8.

4.2 Codeword Interaction

Our goal in this section is to define the criteria an ideal decoder must meet to function properly. We do this using binary codewords, and begin with a description of how codewords interact. Given two NWs, $\boldsymbol{n_a}$ and $\boldsymbol{n_b}$, with codewords $\boldsymbol{c^a}$ and $\boldsymbol{c^b}$, consider the following definitions:

Definition 4.2 *If, for all j, $c^a_j = 1$ when $c^b_j = 1$ we say $\boldsymbol{c^b}$* **implies** *$\boldsymbol{c^a}$, denoted $\boldsymbol{c^b} \Rightarrow \boldsymbol{c^a}$. Any AW that controls $\boldsymbol{n_b}$ also controls $\boldsymbol{n_b}$. It is impossible to turn off $\boldsymbol{n_b}$ without also turning off $\boldsymbol{n_a}$.*

Definition 4.3 *The* **complement** *of codeword $\boldsymbol{c^i}$, denoted $\overline{\boldsymbol{c^i}}$ is the* NOT *of M-bit vector $\boldsymbol{c^i}$. If $\overline{\boldsymbol{c^a}} \cdot \boldsymbol{c^b} = 0$, $\boldsymbol{c^a}$ has a 1 everywhere $\boldsymbol{c^b}$ has a 1, and $\boldsymbol{c^a} \Rightarrow \boldsymbol{c^b}$. If $\boldsymbol{c^a} \not\Rightarrow \boldsymbol{c^b}$, $\overline{\boldsymbol{c^a}} \cdot \boldsymbol{c^b} = 1$, so activation pattern $\boldsymbol{a} = \overline{\boldsymbol{c^a}}$ turns off NW $\boldsymbol{n_a}$, but not $\boldsymbol{n_b}$ (since $\overline{\boldsymbol{c^a}} \cdot \boldsymbol{c^a} = 0$).*

Definition 4.4 *If $\boldsymbol{c^a} \not\Rightarrow \boldsymbol{c^b}$ and $\boldsymbol{c^b} \not\Rightarrow \boldsymbol{c^a}$, NWs $\boldsymbol{n_a}$ and $\boldsymbol{n_b}$ are* **independently controllable**.

Now consider a set, $\mathcal{S}$ of codewords:

Definition 4.5 *$\mathcal{S}$, is* **addressable** *if and only if there exists an activation pattern, $\boldsymbol{a}$, such that exactly the NWs with codewords in $\mathcal{S}$ are turned off (and no other NWs). A particular codeword, $\boldsymbol{c^i}$, is addressable if the set $\{\boldsymbol{c^i}\}$ is addressable. A NW is addressable if its codeword is addressable and no other NW has that codeword.*

We have now defined codeword implication, independent controllability and addressability. The relationships between these concepts are highlighted by the following two lemmas.

Lemma 4.1 *A codeword is addressable if and only if no other codeword implies it. If a set of NWs is addressable, it must contain an addressable codeword.*

Proof Consider a codeword $\boldsymbol{c^a}$. If it is addressable, it is addressed with an activation pattern that turns off all other codewords. Each of these codewords must have a 1 in a position that $\boldsymbol{c^a}$ has a 0, so no other codeword implies $\boldsymbol{c^a}$. Conversely, if no other codewords imply $\boldsymbol{c^a}$, it is addressed with activation pattern $\boldsymbol{a} = \overline{\boldsymbol{c^a}}$.

Now consider the set, $\mathcal{S}$, of codewords present in a set of addressable NWs. Let $\boldsymbol{a}$ be the activation pattern that addresses $\mathcal{S}$. To show that an addressable codeword exists in $\mathcal{S}$, modify $\boldsymbol{a}$ by activating an additional AW that does not turn off every codeword in $\mathcal{S}$ (if there is no such AW, there is only one codeword in $\mathcal{S}$). Remove from $\mathcal{S}$ every codeword that was turned off, then repeat until only one codeword remains. This codeword is addressable. ■

Lemma 4.2 *If two codewords, $\boldsymbol{c^a}$ and $\boldsymbol{c^b}$ are addressable, they are independently controllable. If all codewords are independently controllable, they are all addressable.*

Proof If $\boldsymbol{c^a}$ and $\boldsymbol{c^b}$ are addressable, Lemma 4.1 implies that $\boldsymbol{c^a} \not\Rightarrow \boldsymbol{c^b}$ and $\boldsymbol{c^b} \not\Rightarrow \boldsymbol{c^a}$, so the codewords are independently controllable. If all codewords are independently controllable, then no codeword is implied by any other codeword, so Lemma 4.1 implies that all codewords are addressable. ■

4.3 Ideal Decoders for Memories

NW decoders can be used to control a memory (as in Figure 1). In a crossbar-based memory, a composite decoder is employed along each dimension of the crossbar to select a set of crosspoints. If each of these decoders can each turn on D distinct subsets of NWs, they can collectively activate D^2 sets of crosspoints.

If bits in memory are not written to overlapping locations, the D^2 sets of crosspoints must be disjoint. This implies that the D sets of NWs activated by each decoder must be disjoint, which in turn suggests the following definition and lemma.

A simple NW decoder which can address D disjoint sets of NWs is called a **D-address simple memory decoder**. Its codewords must meet the following requirement.

Lemma 4.3 *A decoder is a D-address simple memory decoder if and only if there exist D addressable codewords.*

Proof If D addressable codewords exist, the NWs with these codewords form D disjoint addressable subsets. If D disjoint subsets of NWs exist, Lemma 4.1 implies that each contains an addressable codeword. Since each of these codewords came from a disjoint addressable subset, these D addressable codewords are distinct. ■

Now consider a composite decoder which contains g simple decoders. If the i^{th} simple decoder in a composite NW decoder is a D_i-address simple decoder, than the composite decoder can address $\sum_{i=1}^{g} D_i$

disjoint subsets of NWs. In Section 5 we compute the minimum codeword length a D-address simple decoder requires.

4.4 Ideal Decoders for Circuits

A memory decoder must activate disjoint subsets of wires in order to control a memory efficiently. We now consider a decoder that controls inputs to a circuit. As we show, the requirements on such a decoder are significantly increased.

If the circuit has D inputs, the decoder must provide control over a set, S, of D NWs. To supply the circuit with all 2^D possible inputs, the decoder must be able to address every subset of S with respect to S. We call such a decoder a **D-address circuit decoder** and give a condition on its codewords.

Definition 4.6 *Let S be a set of NWs that contains NW n_i. n_i is* **uniquely controllable** *with respect to S if there exists a j such that $c_j^i = 1$ and $c_j^a = 0$ for every other $n_a \in S$. Here a_j* **uniquely controls** *n_i, with respect to S.*

Lemma 4.4 *A decoder is a D-address circuit decoder if and only if there exists a set S of size D such that each NW in S is uniquely controllable with respect to S.*

Proof If every NW in S is uniquely controllable, there is an a_j which uniquely controls each $n_i \in S$. To turn off a subset, S' of NWs, set $a_j = 1$ if and only if a_j uniquely controls a NW in S'. For the converse, assume each subset of S is addressable (ignoring NWs not in S). Since each set $S - \{n_i\}$ is addressable, each NW n_i is uniquely controllable with respect to S. ■

Our lemma does not assume a simple NW decoder. The condition on S still holds even if the decoder is composite. Composite decoders are very useful for controlling a memory, but do not ease the requirements for controlling a circuit. Controlling N NW inputs to a circuit requires a simple decoder with at least N AW inputs. If decoders are assembled stochastically, it may require many more than N inputs, as discussed in Section 8. This suggests an I/O challenge for nanoscale architectures.

5 Minimum Code Length

Even when codeword assignment is stochastic, some decoders (axial and radial, for example) provide substantial control over which codewords are generated. Also, future nanotechnology may permit codewords to be deterministically programmed. In both cases, it is desirable to satisfy the conditions established Lemmas 4.3 and 4.4 using minimum length codewords. In this section we describe the minimum length codes for memories and circuits.

5.1 Codewords for Circuits

In a circuit decoder, Lemma 4.4 requires a set of D uniquely controllable NWs. Each wire must have a distinct codeword and each of these codewords must have a 1 in a unique position. The lemma implies that M, the number of bits in each codeword, is at least D. $M = D$ if and only if each of the D codewords is drawn from a $(1, D)$-hot code. A **(k, M)-hot code** [11] consists of all binary strings of length M with exactly k 1s.

5.2 Codewords for Memories

In a simple memory decoder, Lemma 4.3 requires a set of D NWs with distinct, addressable codewords. Given M AWs, we now consider how to generate sets of addressable codewords.

First consider binary reflected codes, introduced in [12]. A **binary reflected code** (BRC) is a code that contains all length-M codewords of the form $\boldsymbol{x}\overline{\boldsymbol{x}}$, where $\boldsymbol{x}$ is an arbitrary binary vector (and M is even).

In our previous work, we have found it convenient to use BRCs because each codeword directly corresponds to a binary sequence, $\boldsymbol{x}$. (They also have the property that they are closed under cyclic shift, which models misalignment of axial codes.) BRCs and their subsets have been used by others for the same reason [18, 19].

A BRC contains $2^{M/2}$ codewords. All pairs of codewords are independently addressable (since no codeword implies another codeword). Unfortunately, this is not optimal, as it requires more AWs than some other codes with the same number of addressable codewords.

A (k, M)-hot code contains $\binom{M}{k}$ codewords. As with BRCs, each codeword is addressable. (Also like BRCs, they are closed under cyclic shift.) An $(\lceil M/2 \rceil, M)$-hot code has $\binom{M}{\lceil M/2 \rceil}$ addressable codewords. Since a length M BRC is a subset of the $(M/2, M)$-hot code, $\binom{M}{\lceil M/2 \rceil}$ is clearly greater than $2^{M/2}$. We now prove that $\binom{M}{\lceil M/2 \rceil}$ is optimal.

Lemma 5.1 *Consider a set of C addressable codewords. The set consisting of the complement of each codeword also contains C addressable codewords.*

Proof By Lemma 4.2 all codewords are addressable if and only if for any pair of codewords, $\boldsymbol{c^a} \not\Rightarrow \boldsymbol{c^b}$ and $\boldsymbol{c^b} \not\Rightarrow \boldsymbol{c^a}$. This implies that $\overline{\boldsymbol{c^b}} \not\Rightarrow \overline{\boldsymbol{c^a}}$ and $\overline{\boldsymbol{c^a}} \not\Rightarrow \overline{\boldsymbol{c^b}}$, so the compliment of each pair of codewords is independently controllable. By lemma 4.2, this implies that each of the C compliment codewords is addressable. ■

Lemma 5.2 *Consider a set of C codewords that are independently addressable. If a minimum weight codeword has weight $w < \lfloor M/2 \rfloor$, there exists a code with C' codewords, such that $C' > C$ and all codewords have weight at least $w + 1$.*

Proof Let $w < \lfloor M/2 \rfloor$ be the weight of the k minimum weight codewords. Remove all k w-weight codewords, then create a set of C' codewords by adding each $(w + 1)$-weight codeword that implies one of the removed codewords. Each of these new codewords was not one of the original C, since none of the original codewords were implied by any of the original k codewords.

Each of the added $(w+1)$-weight codewords is implied by at most $w+1$ of the k removed codewords. Each of the k removed codewords implies $M - w$ $(w + 1)$-weight codewords. As a result, at least $k(M - w)/(w + 1)$ codewords were added. Since $2w < M - 1$, $(M - w)/(w + 1) > 1$, and the number of codewords added is greater than k. C' is larger than C.

To see that the C' codewords are addressable, First observe that none of the newly added $(w+1)$-weight codewords are implied by any of the $C' - 1$ other codewords, because all of the C' codewords have weight at least $(w + 1)$. Second, observe that none of the newly added codewords imply any of the other $(w + 1)$ weight codewords. Finally, none of the newly added codewords imply any of the codewords with weight greater than $(w + 1)$, because if one did imply some codeword, it would imply that one of the k removed codewords also implied that codeword. This would contradict the assumption that all of the C original codewords was addressable. ■

Lemma 5.3 *Consider a set of C codewords which are independently addressable. If all codewords have weight $\lfloor M/2 \rfloor$ or $\lceil M/2 \rceil$, there exists a code with $C' \geq C$ codewords such that all codewords have weight $\lceil M/2 \rceil$.*

Proof If M is odd, consider the same replacement described in the proof of Lemma 5.2. As before, C' new addressable codewords are created. Also as before, at least $k(M-w)/w$ codewords were added after k codewords were removed. Unlike before, $2w = M - 1$, so $(M - w)/(w + 1) = 1$. Since at least k codewords are added, $C' \geq C$. ■

Theorem 5.1 *Given M AWs, there exist at most $\binom{M}{\lceil M/2 \rceil}$ addressable codewords.*

Proof Consider a code that maximizes the number of codewords. Lemma 5.1 states that the complement of this code also maximizes the number of codewords.

Lemma 5.2 implies that the code and its complement both have minimum weight codewords of weight at least $\lfloor M/2 \rfloor$. This means that all codewords in either code have weight $\lfloor M/2 \rfloor$ or $\lceil M/2 \rceil$.

Lemma 5.3 states that an equal size code exists where all codewords have weight $\lceil M/2 \rceil$. There are thus at most $\binom{M}{\lceil M/2 \rceil}$ codewords. ■

Corollary 5.1 *For a decoder with M AWs to be a D-address simple decoder $D \leq \binom{M}{\lceil M/2 \rceil}$.*

6 Real-valued codewords

Binary codewords model the behavior of a decoder when NWs behave according to the ideal model described in Section 3.3. This model is clean but not necessarily realistic. Decoding technology based on FETs behaves much closer to the resistive model of NW control. When an AW generates an electric field, the resistance of each NW is increased by some positive amount. A decoder that obeys the resistive model is called a **Resistive Nanowire Decoder**.

We begin with a comparison of the ideal and resistive model by mimicking the analysis of Section 4. This highlights the difficulty of working directly with the resistive model. To overcome this difficulty, we then explain how resistive decoders can be analyzed using the ideal model.

6.1 Real-Valued Codewords

In an ideal decoder, codewords are binary, since an AW either controls a NW or has no effect. In a resistive decoder, codewords are real-valued.

Definition 6.1 *Let η_i be the initial resistance of NW $\boldsymbol{n_i}$ in the absence any influence from AWs. A* **real-valued codeword** *$\boldsymbol{r^i}$ is a length-M vector associated with $\boldsymbol{n_i}$. The j^{th} entry of $\boldsymbol{r^i}$, r^i_j, is the amount by which the j^{th} AW increases the resistance of $\boldsymbol{n_i}$ when carrying a field.*

Although codewords are real-valued in the resistive model, decoder inputs are still binary, each AW is either on or off. Each AW which is on increases each NWs resistance by some amount. When a resistive decoder is supplied with activation pattern, $\boldsymbol{a}$, the resistance of $\boldsymbol{n_i}$ increases by $\sum_{j=1}^{M} a_j \; r^i_j$. Its total resistance is thus $\eta_i + \boldsymbol{a} \cdot \boldsymbol{r^i}$.

6.2 Real-Valued Codeword Interaction

With binary codewords, we were easily able to define terms such as implication and addressability . When a binary codeword is addressed, it remains on while other codewords are turned off. In the resistive model, codewords are never completely off, they merely have a high resistance. As a result, we must quantify implication and addressability with a parameter α.

Definition 6.2 *A set of codewords, $\mathcal{S}$, is* **α-addressable** *if there exists an activation pattern, $\mathbf{a}$, such that the combined resistance of all NWs with codewords not in $\mathcal{S}$ is at least α times that of the resistance of each NW with a codeword in $\mathcal{S}$. As before, a particular codeword, $\boldsymbol{r^i}$, is α-addressable if the set $\{\boldsymbol{r^i}\}$ is addressable. A NW is α-addressable if its codeword is α-addressable and no other NWs have that codeword.*

Definition 6.3 *Given two NWs, $\boldsymbol{n_a}$ and $\boldsymbol{n_b}$ with real-valued codewords $\boldsymbol{r^a}$ and $\boldsymbol{r^b}$, $\boldsymbol{r^b}$* **α-implies** *$\boldsymbol{r^a}$ if, for all activation patterns, the resistance of $\boldsymbol{n_b}$ is less than α times that of $\boldsymbol{n_a}$.*

6.3 Resistive Decoders for Memories

Consider a crossbar-based memory in which a bit is read from the crosspoint defined by perpendicular NWs $\boldsymbol{n_a}$ and $\boldsymbol{n_b}$. As described in Section 2.4, a read operation measures the current that flows from one dimension of the crossbar to the other through diodes with switchable resistances.

If all current that flows from one dimension to the other flows along NWs $\boldsymbol{n_a}$ and $\boldsymbol{n_b}$, it is straightforward to determine the resistance of the diode. If other NWs also carry a current it becomes more difficult to determine the state of the diode. In a resistive decoder, all NWs connected to the same ohmic contact as either $\boldsymbol{n_a}$ or $\boldsymbol{n_b}$ carry some current (since they have a finite resistance). As a result, current flows through the diodes at the crosspoints of these NWs as well. The resistive state of these diodes is unknown.

Consider the two extremes illustrated in Figure 6:

1. The crosspoint between $\boldsymbol{n_a}$ and $\boldsymbol{n_b}$ is set to a high resistance, but all other crosspoints have a low resistance. This maximizes the amount of current that flows from one dimension of the crossbar to the other, given that the crosspoint being read has a high resistance.

2. The crosspoint between $\boldsymbol{n_a}$ and $\boldsymbol{n_b}$ is set to a low resistance, but all other crosspoints have a high resistance. This minimizes the amount of current that flows from one dimension of the crossbar to the other, given that the crosspoint being measured has a low resistance.

A read operation that addresses NWs $\boldsymbol{n_a}$ and $\boldsymbol{n_b}$ must be able to accurately distinguish between these two cases. The current measured in case 1) must be substantially greater than the current measured in case 2). Even if switchable diodes behave perfectly (their high resistance is infinite, their low resistance is zero), the two cases are only distinguishable if the current flowing through each addressed NW is much greater than the total current flowing through all NWs which are not addressed. Equivalently, the combined resistance of the NWs which are not addressed must be much greater than the resistance of the NWs that are addressed. In other words, $\boldsymbol{n_a}$ and $\boldsymbol{n_b}$ must be α-addressable, where $\alpha >> 1$.

Write operations also require that $\alpha >> 1$. A crossbar write operation (described in Section 2.4) applies a potential across one or more NW crosspoints by placing a charge on the relevant NWs. In order to place the charge, each addressed NW must accumulate a charge much more rapidly than any NW which is not addressed. In other words, we require that the NWs be α-addressed where $\alpha >> 1$.

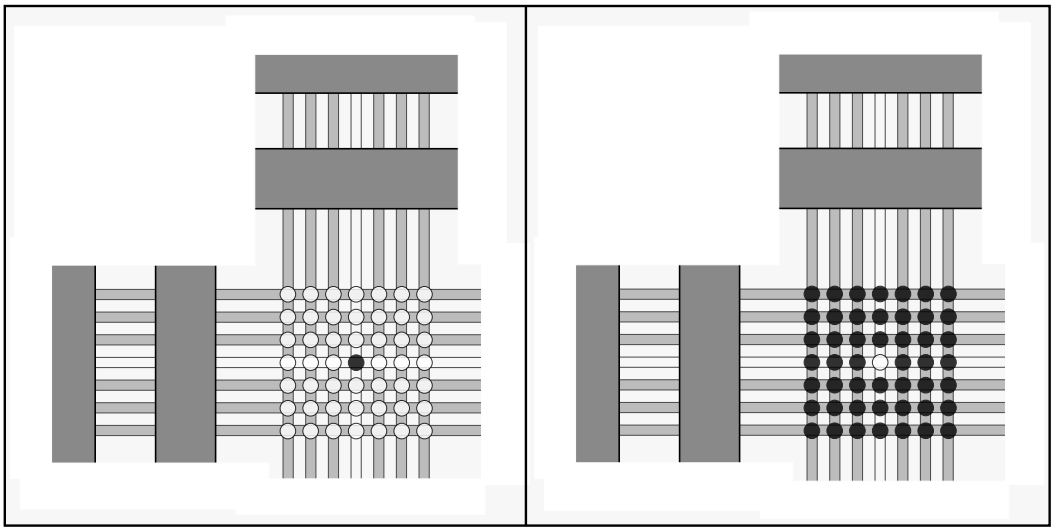

Figure 6. The two extreme cases with respect to a read operation. In the first case, the crosspoint being read has a very high resistance, but all other crosspoints have a low resistance. In the second case, the crosspoint being read has a low resistance, but all other crosspoints have a high resistance. For a crossbar memory to function properly, these two cases must be distinguishable.

Although the addressability condition above is easy to state, it is difficult to work with. Lemma 4.1 states that a binary codeword is addressable if and only if no other codeword implies it. Attempting to provide an equivalent lemma for real-valued codewords yields the following.

Lemma 6.1 *NW $\boldsymbol{n_a}$ is α-addressable only if no codeword α-implies $\boldsymbol{r^a}$. $\boldsymbol{r^a}$, however, is not necessarily α-addressable even if no other codeword $c\alpha$-implies it (where c is some constant).*

Proof If NW $\boldsymbol{n_a}$ is addressable, there exists an activation pattern, $\boldsymbol{a}$, that causes the combined resistance of all other NWs to be at least α times that of $\boldsymbol{n_a}$. Since $\boldsymbol{a}$ causes the resistance of every other NW to be at least α time that of $\boldsymbol{n_a}$, no other NW's codeword implies $\boldsymbol{r^a}$.

Now suppose no other codeword $c\alpha$-implies $\boldsymbol{r^a}$. Consider two other codewords, $\boldsymbol{r^b}$ and $\boldsymbol{r^c}$. Suppose each is identical to $\boldsymbol{r^a}$ except for one entry. Let $r_j^b = c' r_j^a$ and $r_k^c = c' r_k^a$. Also assume that all three NWs have the same initial resistance, η_a, and that $r_j^a < r_k^a$.

Activating only the j^{th} AW causes $\boldsymbol{n_a}$ to have resistance $\eta_a + r_j^a$ and $\boldsymbol{n_b}$ to have resistance $\eta_a + c' r_j^a$. Activating only the k^{th} AW causes $\boldsymbol{n_a}$ to have resistance $\eta_a + r_k^a$ and $\boldsymbol{n_c}$ to have resistance $\eta_a + c' r_k^a$. If $c' = 2c\alpha$ and $\eta_a = r_j^a$, neither $\boldsymbol{r^b}$ nor $\boldsymbol{r^c}$ $c\alpha$-implies $\boldsymbol{r^a}$.

If $\boldsymbol{n_a}$ is addressable, it must be addressed by activating both the j^{th} and k^{th} AWs. When these AWs are activated, the three NWs, $\boldsymbol{n_a}$, $\boldsymbol{n_b}$ and $\boldsymbol{n_c}$ have resistances $2r_j^a + r_k^a$, $(2c\alpha + 1)r_j^a + r_k^a$ and $2r_j^a + 2c\alpha r_k^a$ respectively.

If $r_k^a = d r_j^a$ these resistances become $(d+2)r_j^a$, $(2c\alpha + d + 1)r_j^a$ and $(2c\alpha d + 2)r_j^a$. If $d = 2\alpha c$, $\boldsymbol{n_a}$ is not α-addressable for $\alpha \geq 2$. If d is larger still, $\boldsymbol{n_a}$ is not α-addressable for values of α between 1 and 2. ■

Unlike Lemma 4.1, Lemma 6.1 is not bidirectional. A similar difficulty arises when we consider the resistive equivalent of Lemma 4.3. As we have seen, a D-address resistive memory decoder must contain at least D disjoint subsets of α-addressable NWs, where $\alpha >> 1$. It is sufficient that the decoder have D α-addressable codewords, but not necessary. For example, a resistive memory decoder might α-address the set of NWs $\{n_a, n_b\}$, but if $\boldsymbol{r^a}$ and $\boldsymbol{r^b}$ were very close, but still distinct, neither individual NW would be α-addressable.

6.4 Resistive Conditions for Ideal Decoders

Rather than continue to work directly with real-valued codewords, we described how they can be mapped to binary codewords.

First consider two thresholds, r_{low} and r_{high}, and a simple resistive decoder in which each r_j^i is either less than r_{low} or greater that r_{high}. We now show that this decoder can be treated as ideal if, for some $c > 0$, $r_{high}/r_{low} >> (1+c)MN$, and $\eta_i << cMr_{low}$, for each NWs initial resistance η_i.

Initially each NW $\boldsymbol{n_i}$ has resistance η_i, which is at most r_{high}/N. If the decoder is supplied activation pattern $\boldsymbol{a}$, $\boldsymbol{n_i}$ has resistance $\eta_i + \boldsymbol{a} \cdot \boldsymbol{r^i}$. Since each r_j^i is either less than r_{low} or greater than r_{high}, the resistance of $\boldsymbol{n_i}$ is either greater than $\eta_i + r_{high}$, or less than $cMr_{low} + Mr_{low}$.

The combined resistance of all NWs with resistance at least $\eta_i + r_{high}$ is at least $(\eta_i + r_{high})/N$, since at most N such resistances are combined in parallel. Since $(\eta_i + r_{high})/N > r_{high}/N >> (1+c)Mr_{low}$, the set of NWs with a high resistance (at least $\eta_i + r_{high}$) have a combined resistance that is substantially higher than each wire with low resistance (less than $(1+c)Mr_{low}$). For any $\boldsymbol{a}$, the NWs with low resistance can be considered addressed.

To model a resistive decoder as ideal, we map each real-valued codeword $\boldsymbol{r^i}$ to binary codeword $\boldsymbol{c^i}$ as follows:

1. For some $c > 0$, select r_{low} and r_{high} such that $r_{high} >> (1+c)MNr_{low}$ and $cMr_{low} >> \eta_i$ for all NWs $\boldsymbol{n_i}$.
2. For each $r_j^i > r_{high}$, let $c_j^i = 1$
3. For each $r_j^i < r_{low}$, let $c_j^i = 0$
4. For each $r_{low} < r_j^i < r_{high}$, treat c_j^i as an error.

We have already discussed how ideal decoders can be created from binary codewords in the absence of errors. In the next section, we introduce errors to deal with AWs that provide only partial control over NWs.

7 Binary codewords with errors

The ideal model becomes more practical if it contains the notion of an error. In this section we present the **ideal decoder with errors**, which obeys this modified model. To describe the decoder, we once again use binary codewords, but now allow a codeword bit, c_j^i, to take value e as well as 0 and 1. More precisely:

1. When $c_j^i = 0$, the j^{th} AW provides total control over NW $\boldsymbol{n_i}$. When the AW is on, the NW carries no current.
2. When $c_j^i = 1$, the j^{th} AW provides no control over NW $\boldsymbol{n_i}$. When the AW is on, the NW is unaffected.

3. When $c_j^i = e$, the j^{th} AW provides only partial control over NW $\boldsymbol{n_i}$. When the AW is on, the NWs current decreases by an unknown amount. In this case, c_j^i is called **an error**.

.

If only AWs for which $c_j^i = e$ are activated, $\boldsymbol{n_i}$ will behave unpredictably. To address $\boldsymbol{n_i}$, only NWs for which $c_j^i = 0$ must be activated. To address some other NW, and thus turn off $\boldsymbol{n_i}$, an AW for which $c_j^i = 1$ must be activated. To obtain reliable behavior from an ideal decoder with errors, we must never supply an activation pattern which causes a NW to behave unpredictably.

In the ideal model without errors, lemma 4.3 showed that a D-address simple memory decoder must contain D addressable codewords. We now wish to describe a D-address simple memory decoder that contains D addressable codewords even when up to d codeword bits are corrupted by errors. Such a decoder could tolerate the following types of errors:

1. After codewords are assigned, d bit flips occur. This models etching errors in radial decoders.
2. d bits are partially corrupted, no longer behaving as 0s or 1s. This models the partial control that may result from a faulty resistive decoder. Partial control could result from misalignment, as well as variations in dopant or spacial separation between NWs and AWs.
3. d transient errors in which a 1 becomes a 0 or is partially corrupted.

Unfortunately, transient errors in which a 0 becomes a 1 (and a NW is turned off incorrectly) cannot be tolerated because even one such error can prevent a corrupted NW from being addressed. Fortunately, these errors may be detectable by measuring the current produced by the decoder.

In the absence of errors, two NWs, $\boldsymbol{n_a}$ and $\boldsymbol{n_b}$ are independently controllable in the if $\boldsymbol{c^a} \not\Rightarrow \boldsymbol{c^b}$ and $\boldsymbol{c^b} \not\Rightarrow \boldsymbol{c^a}$. A NW, $\boldsymbol{n_a}$, is addressable if $\boldsymbol{c^i} \not\Rightarrow \boldsymbol{c^a}$ for every other NW $\boldsymbol{n_i}$. To account for errors, we generalize these terms as follows.

Definition 7.1 *Two NWs are* d**-independently controllable** *if they remain independently addressable when up to* d *bit flips occur in their codewords. A NW is* d**-addressable** *if it is* d*-independently controllable with every other NW.*

NWs which are d-independently controllable will be able to tolerate up to d errors. To characterize the codewords of these NWs, we define a measure of distance between codewords.

Definition 7.2 *Let* $/\boldsymbol{c^a} - \boldsymbol{c^b}/$ *be the number of* j *for which* $c_j^a = 1$ *but* $c_j^b = 0$*. The* **balanced Hamming distance** *between* $\boldsymbol{c^a}$ *and* $\boldsymbol{c^b}$ *is* $2\min\left(/\boldsymbol{c^a} - \boldsymbol{c^b}/, /\boldsymbol{c^b} - \boldsymbol{c^a}/\right)$.

The value of this definition is demonstrated by the following lemma.

Lemma 7.1 *If all pairs of codewords have a balanced Hamming distance of at least* $2d + 2$*, they are* d*-independently controllable. In this case, a total of* d *errors can be corrected. Also, up to* $d/2$ *errors in every codeword can be corrected.*

Proof If two codewords are not d-independently controllable, there exists a sequence of d bit flips after which one codeword, $\boldsymbol{c^a}$, implies the other codeword, $\boldsymbol{c^b}$. If $\boldsymbol{c^a}$ implies $\boldsymbol{c^b}$ then $/\boldsymbol{c^a} - \boldsymbol{c^b}/ = 0$. Since each bit flip reduces $/\boldsymbol{c^a} - \boldsymbol{c^b}/$ by at most 1, $/\boldsymbol{c^a} - \boldsymbol{c^b}/$ is at most d initially. Two codewords that are not d-independently controllable thus have balanced Hamming distance at most $2d$.

If all pairs of codewords are d-independently controllable, any pair of codewords remains independently controllable if d bit flips occur. Similarly, the pair of codewords remains independently controllable if $d/2$ bit flips occur in each codeword. ■

Lemma 7.2 *If codewords have a balanced Hamming distance of* $2d+2$*, they have a normal Hamming distance of at least* $2d+2$*. Two BRC codewords,* $\boldsymbol{x_1\overline{x_1}}$ *and* $\boldsymbol{x_2\overline{x_2}}$ *have a balanced Hamming distance* $2d+2$ *if and only if* $\boldsymbol{x_1}$ *and* $\boldsymbol{\overline{x_2}}$ *have a normal Hamming distance* $d+1$*.*

Proof The Hamming distance between two codewords, $|\boldsymbol{c^a}-\boldsymbol{c^b}|$, equals the sum of their directed distances, $/\boldsymbol{c^a}-\boldsymbol{c^b}/+/\boldsymbol{c^b}-\boldsymbol{c^a}/$. Since $/\boldsymbol{c^a}-\boldsymbol{c^b}/+/\boldsymbol{c^b}-\boldsymbol{c^a}/ \geq 2\min(/\boldsymbol{c^a}-\boldsymbol{c^b}/, /\boldsymbol{c^b}-\boldsymbol{c^a}/)$, two codewords with a balanced Hamming distance of $2d+2$ have a normal Hamming distance of at least $2d+2$.

In the case of BRC codewords, $/\boldsymbol{x_1\overline{x_1}}-\boldsymbol{x_2\overline{x_2}}/ = /\boldsymbol{x_1}-\boldsymbol{x_2}/+/\boldsymbol{\overline{x_1}}-\boldsymbol{\overline{x_2}}/$. Since $/\boldsymbol{\overline{x_1}}-\boldsymbol{\overline{x_2}}/ = /\boldsymbol{x_2}-\boldsymbol{x_1}/$, $/\boldsymbol{x_1\overline{x_1}}-\boldsymbol{x_2\overline{x_2}}/ = |\boldsymbol{x_1}-\boldsymbol{x_2}|$. This implies that $/\boldsymbol{x_1\overline{x_1}}-\boldsymbol{x_2\overline{x_2}}/ = /\boldsymbol{x_2\overline{x_2}}-\boldsymbol{x_1\overline{x_1}}/$, so the balanced Hamming distance between the two BRC codewords is $2/\boldsymbol{x_1\overline{x_1}}-\boldsymbol{x_2\overline{x_2}}/ = 2|\boldsymbol{x_1}-\boldsymbol{x_2}|$. ■

The last observation allows the x_i to be codewords from a standard error correcting code. Previous work on fault tolerant NW decoders assumes binary reflected codewords [18] and thus fails to define the more general notion of balanced Hamming distance. This work also does not accurately categorize which codewords are addressable. If errors cause $\boldsymbol{c^a} \Rightarrow \boldsymbol{c^b}$, the authors assume neither codeword is addressable. Lemma 4.1 corrects this assumption.

8 Stochastic Assembly

As described in Section 2, decoders are either assembled stochastically, or programmed using a stochastically assembled decoder. In the latter case, a stochastically assembled memory decoder is required to address individual NWs for programming. In both cases, a properly functioning decoder must be assembled with high probability.

In a stochastically assembled decoder, codewords are assigned to NWs according to some distribution. Different manufacturing processes provide different degrees of control over the distribution of codewords. Radial, axial, masked-based and random contact decoders all allow us specify the number of AWs used, and thus the length of codewords. Radial and axial decoders also offer significant control over which codewords of a given length are produced.

In this section, we review some of our results concerning the number of AWs and codewords required by various types of stochastically assembled decoders. At the end of this section, we discuss how our results translate into area requirements for these decoders.

8.1 Masked-Based Decoders

In a masked-based decoder, lithographically produced high-K dielectric regions are used to couple AWs to subsets of the NWs. When a high-K dielectric region is under an AW, it allows that AWs to turn off the NWs under that region. Since lithographically produced regions are large, they always cover multiple NWs. We use w to denote the number of NWs covered by the smallest high-K dielectric regions that can be reliably produces.

In [25] we bound the number of AWs required to produce a compound mask-based memory decoder in which each simple decoder controls a set of w NWs. We observe that each NW in a set of w NWs is

addressable if and only if a right and left boundary of some high-K dielectric region falls between each pair of consecutive NWs. Since high-K dielectric regions cannot be placed with nanoscale precision, the locations of their endpoints vary from their intended location according to some distribution.

The mask used to produce high-K dielectric regions allows us to target a specific location for each region boundary. Unfortunately, mask manufacture, mask placement and mask application all introduce sources of random variation. We must model this variation to bound the number of dielectric regions required to ensure that region boundaries are present between each pair of NWs with high probability. To accomplish this, we introduced several variants on the standard coupon collector problem. Our main variant, the "targeted coupon collector problem", allows each trial to target a specific coupon. If a trial targets coupon i, the probability of collecting coupon j is a function of $(i-j)$, where subtraction is over the integers $\bmod C$, the number of coupons.

Using a modified version of the targeted coupon collector problem, we are able to show that the number of AWs required to control all w NWs with probability $(1-\epsilon)$ is $\Theta(w\log(w/\epsilon))$. Since AWs are approximately w times larger than NWs, this is not a promising result. Interestingly, $\Theta(w\log(w/\epsilon))$ AWs are also required if each boundary location is chosen uniformly at random over the region of w NWs. The ability to aim region boundaries to within a few NWs, it turns out, does not help significantly when trying to control all NWs.

8.2 Radial and Axial Encoded Nanowire Decoders

In a mask-based decoder, NW codewords are determined by the locations of high-K dielectric regions. Since each region covers many consecutive NWs, adjacent NWs tend to have codewords with 1s and 0s in the same locations. This results in many AWs being required to ensure that all pairs of NWs are independently controllable with high probability.

In contrast, codewords in axial and radial decoders are determined by a NW's encoding. When a decoder is produced, each NW's encoding (and thus its codeword) is independently selected from a large ensemble. Properly chosen encodings ensure that NWs are assigned codewords from a BRC or (h,k)-hot code [12, 27]. Properly chosen concentrations of encodings ensure that each codeword is equally likely to appear on a NW [12].

When a BRC or (h,k)-hot code is used, NWs with distinct codewords are independently controllable. Thus, all w NWs are addressable if they each have a distinct codeword. As shown in [12],

Theorem 8.1 *If w NWs are independently assigned one of C codewords with equal probability, all NWs have distinct codewords with probability $(1-\epsilon)$ when*

$$C \geq w(w-1)/(-2\ln(1-\epsilon))$$

In the case of a BRC, $C = w(w-1)/(-2\ln(1-\epsilon))$ codewords requires $2\lceil\log_2 C\rceil < \lceil 4\log_2(w/2\epsilon)\rceil$ AWs, a substantial improvement over mask-based decoders.

Unfortunately, $C = w(w-1)/(-2\ln(1-\epsilon))$ codewords require that many different NW encodings be produced. In the case of radially encoded NWs, this requires that NWs have many shells, which forces them to be large. To reduce the number of encodings, we bound the number of codewords required for more than $w/2$ codewords to be present with high probability [12].

Theorem 8.2 *If w NWs are independently assigned one of C codewords with equal probability, at least $w/2+1$ codewords are present with probability $(1-\epsilon)$ when*

$$C \geq (w/2)e^{(w-2\ln\epsilon)/(w+1)}$$

If at least $w/2+1$ codewords are present, a $(w/2+1)$-address memory decoder can still be constructed. Although some NWs are wasted, the resulting memory density is still much higher than what can be achieved using only lithographically produced wires [12]. Furthermore, the number of encodings required is proportional to w, the number of NWs connected to each pair of ohmic contacts. When $w = 10$, Monte Carlo simulations have indicate that as few as 12 codewords are required for more than half of all NWs to be addressable [22].

Lemmas 7.1 and 7.2 show that axial and radial decoders become fault-tolerant if each binary reflected codeword, $\boldsymbol{x_i}\overline{\boldsymbol{x_i}}$, is chosen so that each $\boldsymbol{x_i}$ comes from a standard error correcting code. In [24] we show that the etching sequences used to expose NW cores in a radial decoder can be modified to generate such codewords.

8.3 Random Contact Decoders

Like axial and radial decoders, random contact decoders can be made fault tolerant by generating codewords with balanced hamming distance $2d+2$ with high probability [23].

Theorem 8.3 *All N NWs in a randomized contact decoder (where $P(c_j^i = 1) = p$) are d-addresable with probability at least $1-\epsilon$ when the number of AWs M satisfies*

$$M \geq \frac{\left(d + \sqrt{d^2 + 4\ln(N^2/\epsilon)}\right)^2}{4p(1-p)}$$

The bound is $M \geq \ln(N/\sqrt{\epsilon})/(2p(1-p))$ when $d = 0$.

The term "random contact decoder" describes any decoder in which codeword bits have been assigned independently at random. Section 2, suggested several ways in which these decoders could be produced. Creating a fault-tolerant randomized contact decoder does not require precise control over nanoscale features. As such, it appears to be a good candidate for controlling first-generation nanotechnology.

8.4 A General Bound on Circuit Decoders

So far, we have given bounds on the number of AWs required for various types of memory decoders. Now we present a general bound derived for circuit decoders [23]. The conditions of our theorem (given below) apply to axial and radial decoders, as well as random contact decoders. Though the conditions are not met for masked-based decoders, these decoders cannot act as circuit decoders. The large size of each high-K dielectric regions prevent any NW from ever being uniquely controllable, a necessary requirement for a circuit decoder, as shown by lemma 4.4.

Theorem 8.4 *Consider a NW decoder in which all N NWs are uniquely controllable using M AWs with probability $(1-\epsilon)$. If NWs are assigned codewords independently and with fixed probability (meaning no one codeword is biased toward appearing on a particular NW),*

$$M \geq e(1-\epsilon)(N-1)$$

Since the number of AWs required for a circuit decoder is large, multiple simple circuit decoders will likely have to share a common set of mesoscale AWs. This would prevent all of the decoders from being used simultaneously. This suggests an I/O challenge for nanoscale architectures. Even though many nanoscale circuit components may fit into a small area, the inputs to each component may need to be supplied sequentially.

8.5 Codespace Size and Address Translation Circuitry

A stochastically assembled nanoscale architecture must still provide a deterministic interface to the outside world. In the case of memories, bits are accessed using binary addresses. These addresses must be mapped to NW crosspoints, which requires knowledge of what codewords are present on NWs.

When a memory is supplied with a particular binary address, **address translation circuitry** along each dimension of the crossbar maps that address to one or more simple decoders, which are turned on, and an activation pattern, which is supplied as input. To ensure the activation pattern addresses some NW, the address translation circuitry needs information about which codewords are present. In the next section, we discuss how this information can be obtained. For now, assume we can determine the codeword of every addressable NW and consider the storage overhead required.

The behavior of the address translation circuitry depends on the codewords present in each simple decoder. When there are many possible combinations of codewords that can be present, a substantial amount of space must be devoted to storing each codeword. For example if we want all NWs in an axial or radial simple decoder to be addressable, Theorem 8.1 tells us that the number of codewords, C, must be proportional to the number of NWs, w, squared. To record which subset of codewords are present, $\Theta(w \log w)$ bits are required per simple decoder.

In order to make address translation circuitry fast, reliable, and easy to manufacture, it should be implemented in CMOS. Any approximation of its area should take in to account not just the area of AWs and ohmic contacts, but also the area used to store NW codewords. We explicitly model the size of address translation circuitry in [12], but it has received less attention elsewhere. In [10], for example, the net density of crossbar-based memory is estimated, but no area is allotted to translation circuitry.

In general, there appears to be a trade off between the number of NWs one wishes to address, and the amount of area required to successfully address these NWs. Theorem 8.2 showed that the the number of codewords required for axial and radial decoders is substantially reduced when only half of all NWs are addressable. This not only reduces the number of NW encodings that must be manufactured, but also the area of the address translation circuitry used to control the nanoarray.

9 Codeword Discovery

In each stochastically assembled simple decoder, it is necessary to determine which NW codewords are present. This section explains how this is accomplished.

9.1 Binary Codeword Discovery

Our previous work on codeword discovery focused on memory decoders, using read/write operations to determine which NW codewords are present [12]. The read/write operations rely on a crossbar memory's programmable NW crosspoints. In a circuit architecture these might not be present. Also, read/write operations may be slow, and even unreliable. Here we show how codewords can be discovered simply by measuring the current between a decoder's ohmic contacts.

First consider an exhaustive search over all activation patterns in an ideal, fault free decoder. Each activation pattern either turns off all NWs, or addresses some subset of NWs. In the first case, no current flows between ohmic contacts, in the second case current flows. We assume that for each activation pattern, the decoder can detect if current flows, and thus if some subset of NWs are being addressed. To interpret the results of an exhaustive search consider, the following definition and lemma.

Definition 9.1 *Given two activation patterns $\boldsymbol{a}$ and $\boldsymbol{a'}$, $\boldsymbol{a}$* **contains** *$\boldsymbol{a'}$ if a_j = 1 when a'_j = 1.*

Notice that if $\boldsymbol{a'}$ contains $\boldsymbol{a}$, any NW turned off by $\boldsymbol{a}$ is also turned off by $\boldsymbol{a'}$.

Lemma 9.1 *Codeword $\boldsymbol{c^a}$ is present and addressable if and only if $\boldsymbol{a} = \overline{\boldsymbol{c^a}}$ addresses some subset of NWs and no other activation pattern which contains $\boldsymbol{a}$ addresses any NWs.*

Proof If codeword $\boldsymbol{c^a}$ is addressable, it is the only codeword addressed by activation pattern $\boldsymbol{a} = \overline{\boldsymbol{c^a}}$. Every other activation pattern that contains $\boldsymbol{a}$ turns off all NWs that $\boldsymbol{a}$ turns off, as well as all NWs with codeword $\boldsymbol{c^a}$. Thus every other activation pattern that contains $\boldsymbol{a}$ turns off all NWs.

For the converse, observe that if $\boldsymbol{a}$ addresses some subset NWs they must each have either codeword $\boldsymbol{c^a}$ or a codeword that implies $\boldsymbol{c^a}$. Any NWs that have a codeword that implies $\boldsymbol{c^a}$ are also not turned off by some activation pattern that contains $\boldsymbol{a}$. If no activation pattern that contains $\boldsymbol{a}$ addresses any NWs, all NWs addressed by $\boldsymbol{a}$ have codeword $\boldsymbol{c^a}$. ■

In order to determine which addressable codewords are present, we need only identify those activation patterns that address NWs, but which are contained by no activation pattern that addresses NWs. To do this, we can test every activation pattern, and for those that produce a current, check if each of the unactivated AWs can turn off the current. If each can, the compliment of the activation pattern is a codeword. This exhaustive search takes time $O(M2^M)$.

If codewords are drawn from a BRC, an exhaustive is not necessary. In a BRC codewords are of the form $\boldsymbol{x_x}\overline{\boldsymbol{x_x}}$. In this case, activating only the 1^{st} AW tests whether some NW has a codeword that begins with 0. Activating only the $(M/2+1)^{th}$ AW tests whether some NW has a codeword that begins with 1. Activating either the 2^{nd} or $(M/2+2)^{th}$ AW in conjunction with either the 1^{st} or $(M/2+1)^{th}$ tests if any NW has a codeword that begins with 00, 01, 10 or 11 respectively.

More generally, AWs 1 to b and $M/2+1$ to $M/2+b$ can be used to test for codewords that begin with a particular sequence of b bits. This allows us to conduct a binary style search over the codespace. To see how, imagine all $2^{M/2}$ codewords as leaves in a complete binary search tree. For each non-leaf node in the tree, we can test if any codeword below that node is present. We can thus conduct a depth first search through the tree and find w codewords in time $O(w \log M)$.

In an ideal NW decoder, an activation pattern either produces a current or not. In this way, each test in a codeword discovery algorithm produces a binary output. Unfortunately, codeword errors violate this assumption. If an AW only partially turns off some NWs, it may not be possible to determine whether a particularly activation pattern addresses some NW, or just partially addresses multiple NWs.

In order to discover codewords in an ideal NW decoder with errors, we could assume that, for a given activation pattern, the decoder can measure the total amount of current flowing between the two ohmic contacts. Even so, it remains open how to test for sets of addressable NWs in the presence of errors.

10 Conclusion

Nanoscale architectures must be able to control individual NWs. In this paper, we have provided a general model for NW decoders that applies to a wide range of current NW and NW control technologies. In order to ensure our model's generality, we have considered three forms of AW/NW control. In the ideal case, we use binary codewords to provide simple requirements that NW decoders must meet. In the resistive case, we generalize these requirements using real-valued codewords. Since real-valued codewords are difficult to work with directly, we relate them to binary codewords by introducing the notion of errors.

In an ideal decoder with errors, NW codewords are binary, but some bits are corrupted by errors. Our definition of balanced Hamming distance allows us to characterize NW decoders that function correctly even when errors occur. We show that fault-tolerant radial, axial and random contact decoders can all be assembled stochastically. In all cases, a relatively small amount of redundancy allows us to mask errors. This is analogous to how error correcting codes transmit information over a noisy channel. Similar forms of fault-tolerance may apply to other aspects of nanoscale architectures.

Stochastic assembly and fault-tolerance will almost certainly remain key issues in the field of computational nanotechnology. In a stochastically assembled decoder, the location of nanoscale features can vary significantly. As a result, programmable address translation circuitry is required to store which codewords are present. As the variability of codewords increases, so does the size of address translation circuitry. This may hint at a more general tradeoff in stochastically assembled architectures.

A significant I/O challenge that faces nanoscale architectures. Large lithographically produced AWs must provide reliable control over many NWs. Though we have described many promising approaches, the problem of NW control should not be taken lightly. In the case of circuit decoders, for example, we have shown that a large number of AWs are required. If inputs to a nanoscale circuit are not supplied sequentially, the area of AWs, not NWs, limits the number of inputs. Even in memories, the interface between lithographic and nanoscale devices may introduce a significant delay. As computational nanotechnology continues to gain prominence, it will motivate revised models of computation and novel approaches to computer design.

References

[1] Robert Beckman, Ezekiel Johnston-Halperin, Yi Luo, Jonathan E. Green, and James R. Heath. Bridging dimensions: Demultiplexing ultrahigh-density nanowire circuits. *Science*, 310:465–468, 2005.

[2] M. T. Björk, B. J. Ohlsson, T. Sass, A. I. Persson, C. Thelander, M. H. Magnusson, K. Deppert, L. R. Wallenberg, and L. Samuelson. One-dimensional steeplechase for electrons realized. *Nano Letters*, 2(2):87–89, 2002.

[3] Yong Chen, Gun-Young Jung, Doublas A. A. Ohlberg, Xuema Li, Duncan R. Stewart, Jon O. Jeppeson, Kent A. Nielson, J. Fraser Stoddart, and R. Stanley Williams. Nanoscale molecular-switch crossbar circuits. *Nanotechnology*, 14:462–468, 2003.

[4] Costel Sorin Cojocaru, Jean Marie Padovani, Travis Wade, Corrado Mandoli, Gerard Jaskierowicz, Jean Eric Wegrowe, Anna Fontcuberta i Morral, and Didier Pribat. Conformal Anodic Oxidation of Aluminum Thin Films. *Nano Letters*, 5(4):675–680, 2005.

[5] C. P. Collier, E. W. Wong, M. Belohradský, F. M. Raymo, J. F. Stoddart, P. J. Kuekes, R. S. Williams, and J. R. Heath. Electronically configurable molecular-based logic gates. *Science*, 285:391–394, 1999.

[6] Charles P. Collier, Gunter Mattersteig, Eric W. Wong, Yi Luo, Kristen Beverly, José Sampaio, Francisco Raymo, J. Fraser Stoddart, and James R. Heath. A [2]catenate-based solid state electronically reconfigurable switch. *Science*, 290:1172–1175, 2000.

[7] André DeHon. Array-based architecture for FET-based, nanoscale electronics. *IEEE Transactions on Nanotechnology*, 2(1):23–32, Mar. 2003.

[8] André Dehon. Deterministic addressing of nanoscale devices assembled at sublithographic pitches. *IEEE Transactions on Nanotechnology*, 4(6):681–687, 2005.

[9] André; DeHon. Nanowire-based programmable architectures. *J. Emerg. Technol. Comput. Syst.*, 1(2):109–162, 2005.

[10] André DeHon, Seth Copen Goldstein, Philip Kuekes, and Patrick Lincoln. Nonphotolithographic nanoscale memory density prospects. *IEEE Transactions on Nanotechnology*, 4(2):215–228, 2005.

[11] André DeHon, Patrick Lincoln, and John E. Savage. Stochastic assembly of sublithographic nanoscale interfaces. *IEEE Transactions on Nanotechnology*, 2(3):165–174, 2003.

[12] Benjamin Gojman, Eric Rachlin, and John E. Savage. Evaluation of design strategies for stochastically assembled nanoarray memories. *J. Emerg. Technol. Comput. Syst.*, 1(2):73–108, 2005.

[13] S. C. Goldstein and M. Budiu. NanoFabrics: spatial computing using molecular electronics. *Procs. 28th Annl. Int. Symp. on Computer Architecture*, pages 178–189, June 2001. see http:// www.cs.cmu.edu/ seth/papers/isca01.pdf.

[14] K. Gopalakrishnan, R. S. Shenoy, C. Rettner, R. King, Y. Zhang, B. Kurdi, L. D. Bozano, J. J. Welser, M. B. Rothwell, M. Jurich, M. I. Sanchez, M. Hernandez, P. M. Rice, W. P. Risk, and H. K. Wickramasinghe. The micro to nano addressing block (mnab). In *2005 IEEE International Electron Devices Meeting*, page 19.4, Washington DC, December 5 - 7 2005.

[15] Mark S. Gudiksen, Lincoln J. Lauhon, Jianfang Wang, David C. Smith, and Charles M. Lieber. Growth of nanowire superlattice structures for nanoscale photonics and electronics. *Nature*, 415:617–620, February 7, 2002.

[16] Tad Hogg and Greg Snider. Defect-tolerant logic with nanoscale crossbar circuits. Technical report, HP Labs, 2004. see http://www.hpl.hp.com/research/idl/papers/molecularAdder/.

[17] E. Johnston-Halperin, R. Beckman, Y. Luo, N. Melosh, J. Green, and J.R. Heath. Fabrication of conducting silicon nanowire arrays. *J. Applied Physics Letters*, 96(10):5921–5923, 2004.

[18] Philip J Kuekes, Warren Robinett, Gabriel Seroussi, and R Stanley Williams. Defect-tolerant interconnect to nanoelectronic circuits. *Nanotechnology*, 16:869–882, 2005.

[19] Philip J Kuekes, Warren Robinett, and R Stanley Williams. Improved voltage margins using linear error correcting-codes in resistor-logic demultiplexers for nanoelectronics. *Nanotechnology*, 16:1419–1432, 2005.

[20] Lincoln J. Lauhon, Mark S. Gudiksen, Deli Wang, and Charles M. Lieber. Epitaxial core-shell and core-multishell nanowire heterostructures. *Nature*, 420:57–61, 2002.

[21] Nicholas A. Melosh, Akram Boukai, Frederic Diana, Brian Gerardot, Antonio Badolato, Pierre M. Petroff, and James R. Heath. Ultrahigh-density nanowire lattices and circuits. *Science*, 300:112–115, Apr. 4, 2003.

[22] Eric Rachlin and John E. Savage. Nanotechnology note: Small codespace addressing strategies for nanoarrays. IC Note 3, Brown University, May 2005.

[23] Eric Rachlin and John E Savage. Nanowire addressing in the face of uncertainty. In *Procs 2006 Int. Symp. on VLSI*, Karlsruhe, Germany, March 2-3, 2006.

[24] Eric Rachlin and John E Savage. Crossbar addressing using core-shell nanowires (extended abstract). In *SRC Techcon 2005*, Portland, Oregon, October 22-24, 2005.

[25] Eric Rachlin, John E Savage, and Benjamin Gojman. Analysis of mask-based nanowire decoders. Technical report, Department of Computer Science, Brown University, 2006. In preparation.

[26] Eric Rachlin, John E Savage, and Benjamin Gojman. Analysis of a mask-based nanowire decoder. In *Procs 2005 Int. Symp. on VLSI*, Tampa, FL, May 11-12, 2005.

[27] John E. Savage, Eric Rachlin, André DeHon, Charles Lieber, and Yue Wu. Radial addressing of nanowires. Technical report, Computer Science Department, Brown University, 2005. To appear in J. Emerg. Technol. Comput. Syst.

[28] D. B. Strukov and K. K. Likharev. Cmol fpga: a reconfigurable architecture for hybrid digital circuits with two-terminal nanodevices. *Nanotechnology*, 16(6):888–900, 2005.

[29] Dongmok Whang, Son Jin, Yue Wu, and C. M. Lieber. Large-scale hierarchical organization of nanowire arrays for integrated nanosystems. *Nano Letters*, 3(9):1255–1259, 2003.

[30] R. S. Williams and P. J. Kuekes. Demultiplexer for a molecular wire crossbar network, US Patent Number 6,256,767, July 3, 2001.

[31] J. Woodruff, J. Ratchford, H. Jagannathan, H. Adhikari, H.-S.P. Wong2, and C.E.D. Chidsey. Deterministic nanowire growth (extended abstract). In *SRC Techcon 2005*, Portland, Oregon, October 22-24, 2005.

[32] Yiying Wu, Rong Fan, and Peidong Yang. Block-by-block growth of single-crystal Si/SiGe superlattice nanowires. *Nano Letters*, 2(2):83–86, 2002.