The theory and practice of Probabilistic CMOS

Master Thesis

Author:
Luuk Oudshoorn Bsc

Supervisors:
dr. ir. A.B.J. Kokkeler
prof. dr. ir. G.J.M. Smit
ir. E. Molenkamp
dr. ir. A.J. Annema

CAES group University of Twente

April 25, 2016

Contents

Abstract

It is expected that in the future noise will become dominant in minimum size CMOS logic, inducing probabilistic behaviour. This allows for a trade off between performance and energy consumption. The trade off can be visualized with an Energy-Probability (E-P) curve, which defines the relationship between energy consumption and probability of correctness. This work presents a more realistic view on probabilistic CMOS behaviour for both isolated logic gates and connected systems.

Theoretically, the probability of correctness of an ideal CMOS system in the presence of noise can be represented by a function of the energy consumed, the so called error function. However, CMOS speed drops rapidly around the threshold voltage. This introduces additional errors when signals miss the setup time deadline at their next sample point. The exact probability at a specific supply voltage level is influenced by factors such as noise levels, operating frequency, transistor size and temperature. Current models of Probabilistic CMOS ignore these imperfections and are therefore unrealistic at low supply voltage levels.

At a system level, designs can be made with Probabilistic CMOS. Similar to conventional deterministic logic, such designs should ideally be made up of separately characterized standard building blocks. Current models of probabilistic systems only model the propagation of noise and errors through a system. In order to get a better model of probabilistic systems, errors due to missed deadlines should be added to the models of probabilistic systems. A missed deadline in this context is a changing signal that does not assume the correct value before the next sample moment. In contrast to conventional digital logic, these propagation delays have a large variance. When a more accurate prediction of the expected total error of a probabilistic system can be made for a specific energy distribution, an algorithm can be developed that finds the optimal energy distribution for a certain total error. For a system with two independent parts an algorithm was developed that can find the optimal energy distribution for low expected errors.

1 Introduction

Just after the second world war, in 1948, British writer George Orwell wrote his famous novel *Nineteen Eighty-Four*. Observations during the second world war had led him to believe that democracy would not survive. He foresaw an omnipresent totalitarian government that could track your every move. Orwell used many new concepts to define this government, the most famous being *Big Brother*, which has since become synonym for a government that abuses power for mass surveillance. Fortunately, the future as predicted by Orwell has not come true (at least not entirely). In fact, Orwell's work may have inspired a different path. Even in 2011, staying clear of a *Nineteen Eighty-Four* type of society was used as an argument against surveillance laws in American politics. We resist it's consequences and turn the prediction into a self-defeating prophecy.

In the semiconductor industry predictions are also written, though they usually aren't published as a novel. Some come in the shape of a technology roadmap and are meant as more of a goal/agreement than a prediction and are often self-fulfilling prophecies. Sometimes future problems are predicted in a paper. In successive papers the problem may then be solved, either temporary or permanently. Such a paper was published by Laszlo B Kish in 2002. He predicted that within 7 years, noise may dominate the performance in minimum size digital logic [1]. Even though that did not come true, his calculations are still valid and predict a significant increase in noise. More effort will have to be put into coping with the noise.

This approach of preventing errors caused by noise is rooted in the classical use of digital circuits. If only a single bit flips, the output is considered incorrect. In some cases however this flipped bit may have little influence on the usefulness of the data. Take for example an audio stream. Though the quality of the audio is degraded, the information it carries (e.g. human speech) may still be received correctly if a few bits are flipped.

What if we took that audio signal and processed it on a system that is dominated by noise? It would be wise to compensate for the noise, but only where compensation is needed. By spending more energy on the critical parts of the message (i.e. the most significant bit of a data word), it is ensured that energy is used where it matters most and saved where a mistake cannot do much damage. The advantages of a system that only spends energy where needed are obvious. But how can such a system be designed? How can it be ensured that the system will make no more than the allowed amount of errors and for the least amount of energy? In order for probabilistic systems to be implemented on a large scale, these design questions need to be answered.

This report presents a basis for understanding probabilistic digital logic. First the theory behind probabilistic digital logic is explained in chapter 2. The fundamental limits in energy saving are discussed, along with current models for probabilistic hardware. An analysis of the performance of PCMOS is made for single elements in chapter 3 and for a PCMOS ripple carry adder in chapter 4.

2 The theory of probabilistic CMOS

Every year CMOS devices become smaller. The improvements are governed by the well known self-fulfilling Moore's law. The growth is exponential, doubling the number of transistors on a die every eighteen months. While the size shrinks, the supply voltage is lowered. It has been lowered substantially over the last decades and is still being lowered. Figure 2.1 shows the expected supply voltage for the next years according to the International Technology Roadmap for Semiconductors (ITRS)[2]. For digital logic, lowering the supply voltage decreases the dynamic power consumption and is therefore a desirable effect. Unfortunately, the supply voltage does not scale with Moore's law, but has a linear trend. If it would be lowered more in order to decrease dynamic power consumption faster, the circuits would become substantially slower.

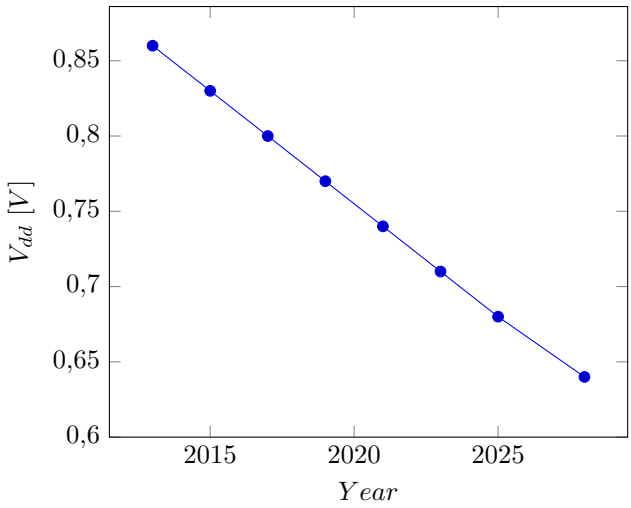


Figure 2.1: The ITRS roadmap for High Performance, high Vdd transistors.

Thermal noise in a transistor depends on the channel resistance and parasitic capacitance (see Figure 2.2 for a model of the parasitic capacitances). When the width and length of the channel are scaled down by the same factor, the channel resistance remains approximately the same because resistance is proportional to the resistor's $\frac{W}{L}$ factor. Capacitance on the other hand is proportional to the gate area, which decreases in both the length and width. Capacitance therefore decreases for smaller feature sizes. This is good news for the speed of MOSFETs, because the channel resistance and parasitic capacitors together form an RC low pass filter that puts an upper limit to the speed of the MOSFET. For noise immunity however this is not good, because less high frequency noise is filtered out. The root mean square (RMS) effective thermal noise voltage across a capacitor is given as[3]:

$$V_n = \sqrt{\frac{k_b T}{C}} \tag{2.1}$$

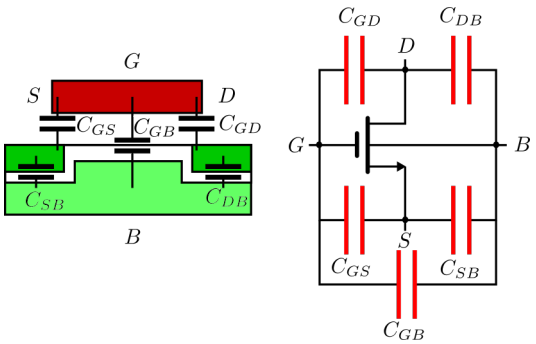


Figure 2.2: Schematic image of a MOSFET with parasitic capacitors. On the left capacitances are drawn over the cross section of a schematical implementation of a MOSFET. On the right the same capacitances are shown in a schematic view.

Where k_b is Boltzmann's constant, T the temperature in Kelvin and C the capacitance. The noise is actually generated by a resistor and filtered by the resistor and capacitor. The resistance value drops out of this equation, making it appear as if the capacitor generates the noise. In the case of a transistor, the channel is the resistor. It generates noise, which is filtered by parasitic capacitors. The effective thermal noise produced by the transistor can therefore be approximated by this equation. In reality, the noise processes are more complex and include a noise efficiency factor (NEF) that describes the relative amount of noise generated by transistors for the same drain current and bandwidth. In this work the simplified equation is used to get a rough estimate of the amount of noise to be expected in the future.

A recent study of field effect transistor dielectrics[4] shows an oxide capacitance in the order of several 5μF per square centimeter for the near future. For a $25nm^2$ transistor (an expected feature size in 15 years[2]) this would result in an effective noise level of 60mV at room temperature for a system without load. While the noise levels are rising over the years, the supply voltage is lowered. This combination has a negative impact on the noise immunity of a transistor[3]. For future technology nodes the effective thermal noise may become too large for digital cmos logic and cause errors[1].

Much effort is put into keeping future digital circuits deterministic in spite of the increasing noise. Methods range from error correcting codes to dynamic voltage scaling [5] or triplication and majority voting. All of these methods however have the disadvantage that additional circuitry is needed. Also, the goal of each solution is to keep the result error free. This is not always necessary. In domains such as audio or radio astronomy, adding a little noise during calculations may be acceptable as long as the amount of noise can be controlled. In 2003 Krishna V. Palem wrote a series of papers that looks into circuits that make errors[6]. Palem showed that there is a fundamentally lower limit to the energy consumption of circuits that work probabilistic instead of deterministic. He proposes to use Probabilistic CMOS (PCMOS) as an alternative for its deterministic counterpart in situations where errors can be tolerated to some degree. In return for allowing an increased amount of noise or even errors, much less energy is consumed by PCMOS.

In this chapter the concept of PCMOS is explained and modeled. The fundamental limits of energy consumption are explored and current models of PCMOS are described. This provides a

basis for further investigation using simulations in the next chapters.

2.1 Thermodynamic theory

2.1.1 Introduction

To understand why probabilistic behavior can lead to fundamentally lower energy consumption, the process of switching needs to be viewed from a thermodynamic perspective. A summary of the reasoning is presented here, for the full explanation the reader is referred to [6].

A thermodynamic system T with N identical elements is considered. An example of such a system is shown in Figure 2.3. It represents a sealed tube with four gas particles. Though they are identical, different colours are given to each in order to distinguish between individual particles.

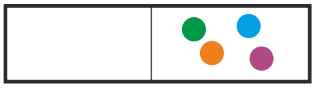


Figure 2.3: A thermodynamic system with four identical elements.

Each particle has a set of so called classical observables. Examples are position and momentum. For this example, we will only consider the property of being in either the left or the right half of the tube. Therefore a total of 16 situations can be distinguished for this system, as shown in Figure 2.4.

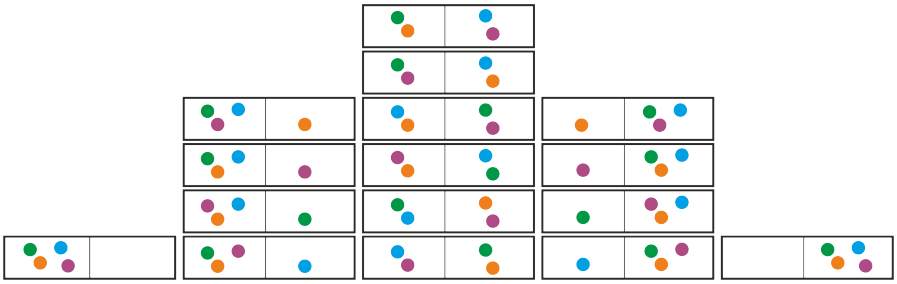


Figure 2.4: 16 uniquely distinguishable states for the thermodynamic system

There are two ways of looking at this system. We are either interested in the microstates or the macrostates. A microstate has information about all the observables of each particle in the system. A unique microstate can be assigned to every possible state of the system. In this very simple example only 16 microstates are possible, but in reality (when position and momentum are classical observables) the number of microstates may be infinite.

In contrast to microstates, macrostates only hold information about the system as a whole. In this example the percentage of particles in the left or right half of the tube is considered. Therefore only 5 macrostates are possible for this system. In more realistic systems the pressure or temperature of a system are examples of macrostates.

An important concept in thermodynamics is entropy. The entropy of a system gives an indication of the amount of disorder. The entropy of a system depends on the macrostate it is in:

$$S = k_b ln\Omega$$

Where k_b is Boltzmann's constant and Ω is the number of microstates for the macrostate it is in. It can be seen in Figure 2.4 that the macrostate with the highest amount of disorder (two particles on each side) has the most microstates and therefore the largest entropy. According to the second law of thermodynamics the entropy of a system can only be decreased by applying energy; without external work the entropy can only increase or remain the same.

2.1.2 Lowering energy consumption

The four particle example system can be used to represent a bit of information (a 1 or 0). The represented value is determined by sampling the location of a random particle. If it is in the left half, a value of zero is chosen. For the right half, a value of one is chosen. If switching between states should be deterministic, the system has to be either in the leftmost or rightmost state shown in Figure 2.4, because in those states all particles represent the correct value. If one of the particles is on the wrong side, there is a probability of 25% that sampling leads to an incorrect value. However, since the entropy for the probabilistic state is higher it will also take less energy to reach that state. In other words, a tradeoff can be chosen between probability of correctness and energy consumption.

Even though this reasoning is relatively easy to follow, the formal proof of this is long and complicated. Palem[6] wrote this proof down in its completeness. The conclusion of the proof is that the minimal energy required for deterministic state switching is $k_b Tln(2)$, while the minimal energy required for probabilistic state switching with probability p is $k_b Tln(2p)$. The potential reduction of energy consumption for probabilistic is the difference between these two: $k_b Tln(p)$.

2.1.3 Representing probabilistic bits

The example of a gas tube works well to explain the terminology and to get an idea of how energy consumption can be lowered for the probabilistic case. In practice however, we represent a bit in the electrical domain. More specifically, the convention is to let 0 volt represent a '0' and the supply voltage represent a '1'. There is usually some tolerance for small deviations of these voltages. For practical reasons this report will assume that an ideal measurement system is used to determine the bit value by taking anything higher than half the supply voltage as a one and anything lower than half the supply voltage as a zero. In reality such a system is very difficult to make, but using it for the models makes every calculation much simpler.

Palem proposed to model a probabilistic bit (PBit) as a white Gaussian noise source around the represented value .Figure 2.5 shows the resulting Probability Density Function (PDF). The probability of measuring the correct value can be found by integrating the part of the PDF which

is on the desired side of half the supply voltage. Integrating a Gaussian PDF results in a so called error function[7]:

$$P(X \leq x) = 0.5 + 0.5\text{erf}(\frac{x-\mu}{\sqrt{2}\sigma}) \tag{2.2}$$

Its properties depend on the noise standard deviation σ and the average voltage level μ (either V_{dd} or zero, depending on the bit). For the supply voltage and noise RMS only positive values are considered. An example of an E-p curve for $\sigma = 0.1$ is shown in Figure 2.6. We are only interested in the probability of measuring a value lower than half the supply voltage when a zero is intended ($\mu = 0$). This probability is the same as the probability of measuring a value higher than half the supply voltage when a one is intended (this can easily be seen from Figure 2.5).

$$p_{correct,0} = P(X \leq \frac{V_{dd}}{2}) = 0.5 + 0.5 \cdot \text{erf}(\frac{0.5 \cdot V_{dd}}{\sqrt{2}\sigma}) \tag{2.3}$$

$$p_{correct,1} = P(X \geq \frac{V_{dd}}{2}) = 0.5 - 0.5 \cdot \text{erf}(\frac{0.5 \cdot V_{dd} - V_{dd}}{\sqrt{2}\sigma}) = p_{correct,0} \tag{2.4}$$

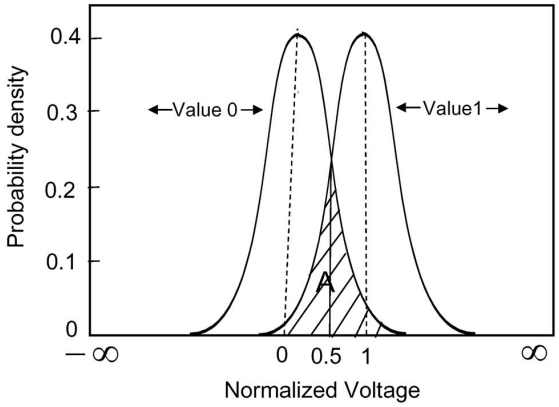


Figure 2.5: Zero and one representations for a probabilistic bit. Source: [8].

Equation 2.2 shows that the probability of correctness at a specific supply voltage depends on the ratio $\frac{V_{dd}}{\sigma}$. A larger effective noise level requires a higher supply voltage level for the same probability. In other words, the probability of correctness at a specific voltage decreases as the noise levels increase. This relation is shown graphically in Figure 2.7.

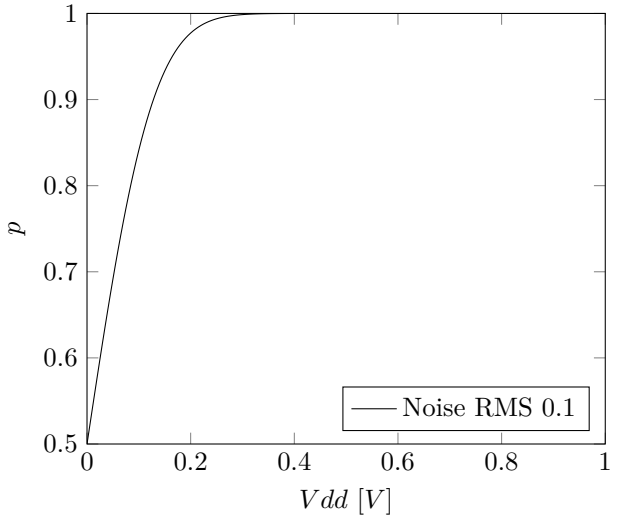


Figure 2.6: Example of an E-p curve for $\sigma = 0.1$.

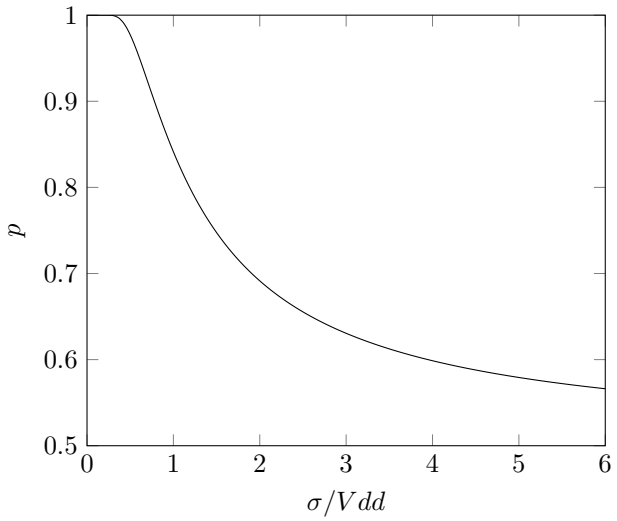


Figure 2.7: Dependency of p on the ratio of noise to supply voltage level.

2.2 Theoretical E-P curves

Up till now, PBits were assumed to simply exist, without giving any thought to the physical process that creates them. In reality, these PBits are created by PCMOS. The PCMOS is part of a circuit that performs a certain task, for example a mathematical multiplication. In order to design such a circuit for a certain probability of error in an efficient way, the relation between probability of correctness and energy consumption is required. The theory provided so far has allowed us to construct a model for representing a PBit using the supply voltage and noise standard deviation as parameters (equation 2.3).

2.2.1 Relating voltage to energy

The supply voltage can be related to the power consumption, which in turn can be integrated over time to get the energy consumed. Power consumption in digital systems can be divided into two parts: Dynamic and static consumption[9]. Static consumption is due to leakage in the transistors and is generally much smaller than the dynamic part, though it is becoming more dominant. The dynamic part of the Power consumption can be approximated by the equation:

$$P_{dynamic} = \mathrm{AC}V_{dd}^2 f \tag{2.5}$$

Where C is the total load capacitance, V_{dd} the supply voltage and f the operating frequency. Not all gates switch every clock cycle. Therefore a compensation factor A is included, which is equal to the average fraction of the circuit that is switching. In most digital systems the output is changed at most once per clock period (though triggering on both flanks of the clock is possible). Therefore charging and then discharging a capacitor takes at least two clock periods. This puts an upper limit of $\frac{1}{2}$ on A, though it will generally be lower because not every output changes every clock period. The factor A depends on circuit usage and will be fixed (or at least be a known random variable). For simplicity the factor A is assumed to be equal to $\frac{1}{2}$ in the rest of this document, indicating a circuit in which all gates are active every clock period.

Lowering the operating frequency results in less power consumption, but not necessarily less energy consumption for the same task. When less operations are performed per time unit, more time is required for the same number of operations. In fact, the total energy for an operation may increase due to the static power consumption (even though it is neglected in approximations here). Changing the load capacitance can be done at design time for a chip, by choosing appropriate gate sizes for transistors. After the chip is produced, the capacitance is mostly fixed. Changes in the supply voltage affect power consumption the most, due to the quadratic relation.

If the f is left out of the expression for dynamic power consumption, equation 2.5 becomes an equation for the average *energy cost* of a single switching operation. It can then be written as:

$$V_{dd} = \sqrt{\frac{2E_{dynamic}}{\mathrm{C}}} \tag{2.6}$$

Which can be used to make an Energy-Probability (E-p) curve from the definition of the PBit (equation 2.3 on page 10):

$$p_{correct} = 0.5 + 0.5 \cdot \mathrm{erf}(\frac{\sqrt{E_{dynamic}}}{2\sigma\sqrt{\mathrm{C}}}) \tag{2.7}$$

When using equation 2.1 from page 6 as the value for σ, this equation reduces to:

$$p_{correct} = 0.5 + 0.5 \cdot \mathrm{erf}(\frac{\sqrt{E_{dynamic}}}{2\sqrt{\mathrm{k_b T}}}) \tag{2.8}$$

Which shows a direct relation between the probability of correctness and power consumption.

2.2.2 Theoretical probability of a PCMOS logic gate

Equation 2.7 relates the energy consumption while calculating a PBit to its probability of correctness. This is done by looking at the PBit as purely being a capacitor that is charged or discharged, still ignoring the hardware that does the actual (dis)charging. In reality a PBit is the output of a PCMOS circuit.

The current way of modeling PCMOS is by simply taking the ideal logical function and applying noise to its output(s). Examples of this can be found in papers from Cheemalavagu[10] (A probabilistic inverter) and Lau[11][12] (resp. a probabilistic ripple carry adder and multiplier). The probability of correctness is calculated by taking the noise into account while performing ideal logical operations. The result is an E-P curve that can be described by an error function. This is supported by a more recent paper by J. Kim[13] on probabilistic circuits.

An assumption in these models is that the hardware still functions correctly at very low supply voltage levels, or that the minimum required supply voltage can be lowered substantially over the next decade. The latter will probably not happen, as the supply voltage roadmap by the ITRS[2] suggests that the supply voltage of minimum size digital logic will only scale down to around 0.7 volts over the next decade. Working with transistors below their threshold voltage can be done, but channel conductance then drops rapidly. Lower conductance results in lower currents for (dis)charging the load capacitor, which therefore takes longer. At a certain supply voltage the circuit will become too slow and starts to miss deadlines. The theoretical models that are presented by Cheemalavagu and Lau therefore give an unrealistic image of the future performance of probabilistic circuits.

3 Performance of a PCMOS inverter

In the previous chapter the existing models of Probabilistic CMOS were explained. These models assume that the PCMOS circuits can function at any supply voltage level and errors are only introduced by the added voltage noise. In this chapter, the influence of reduced speed at lower supply voltage levels is added to this model.

When the gate-source voltage drops below the threshold voltage the transistor goes from saturation to weak inversion and the drain source current drops rapidly. When the drain source current is lowered too much, the load capacitance cannot be (dis)charged in time and the output is interpreted in the wrong way. In a noise free environment (and with an ideal interpreter of a binary one and zero) there is an infinitely small difference between the supply voltage at which the system still makes the deadline and the supply voltage at which it just misses it. If it is too slow, the value of the output is determined by the previous bits (transition from 1 to 1 is easier than from 1 to 0). For an uncorrelated random sequence, the probability of correctness without the influence of noise therefore drops from 1 to 0.5 at one specific supply voltage. Therefore the operating frequency of a system that can be considered noise free should always be reduced when lowering the supply voltage, otherwise the probability of correctness immediately drops to 0.5.

When the noise that is generated by the transistors is taken into account, the transition from 1 to 0.5 probability of correctness is smoothed out over a larger supply voltage range. Due to the random nature of the noise, sometimes it has a positive influence on the desired transition and sometimes a negative influence. Therefore already at voltages higher than that of the noiseless transition from 1 to 0.5, noise will sporadically make the transition slower and introduce errors. Similarly, below the threshold sometimes the noise will have a positive influence on the transition, preventing an error that would otherwise be made. As a result there is a gradual transition from 1 to 0.5. The exact characteristics of the transition are determined by many physical parameters, of which the most important are discussed in this chapter. The goal is to determine how an imperfect system influences the theoretical E-P curves that were presented in the previous chapter. An inverter is chosen as the circuit to be studied, because it is the most simple CMOS circuit possible. In this chapter the influence of other connected circuits is not yet taken into account.

3.1 Simulation

Throughout this chapter and the next, theory is backed up by simulation results. For all the simulations the supply voltage is shown on the horizontal axis of E-P curves. This is done because a voltage can be related directly to MOSFET parameters such as threshold voltage, but is still a good representation of the consumed energy per unit time, as was explained in the previous chapter. A circuit with a lower supply voltage level is always considered to have a lower energy consumption than its counterpart with a higher supply voltage level.

Transistors of the size that we are interested in ($25nm^2$) cannot be made or modeled yet. We want to model smaller transistors because they have larger noise levels. In order to get an idea of how those transistors may behave, an available technology node is simulated with increased noise. The umc65 library is chosen, because it is widely used and available. Cadence IC is used for the simulations. A parameter called *noise scale* is used to multiply every type of noise by a given factor. The 65nm technology with increased noise is used to represent the much smaller future transistors. This does not model the changes in behaviour for future CMOS, but these simulations can be used to analyze the effects of large amounts of noise, which is expected in the future. This model will not provide absolute predictions on performance, but can be used to analyze the qualitative behaviour of such a system.

E-P curves cannot be simulated directly, because the probability of correctness (which is on the vertical axis of the curve) cannot be derived directly from a schematic. The points on the curve are estimated by simulating many bit periods for a supply voltage setting and counting the number of correct and incorrect samples. A long transient simulation is performed using Cadence, which can simulate time domain noise. The results are then exported to Matlab, where the sampling and processing of the data is done. In order to get all the points for the E-P curve, a parameter sweep is performed for the V_{dd} parameter. This generates one E-P curve. If a parameter sweep on another parameter is also desired (generating multiple E-P curves), the number of transient simulations quickly rises. Simulations therefore sometimes take multiple days. In order to keep the simulation time down, the number of points (the supply voltage step size) for the E-P curves is often kept low. This results in a less smooth curve, but also a shorter simulation time.

In each of the following sections of this chapter a design or environment parameter is analyzed and simulated. Four parameters are analyzed in total: Noise amplification, transistor width, operating frequency and temperature. These parameters are analyzed separately in order to isolate their influence. Table 3.1 shows the default parameters for the simulation. The four bold parameters are the ones that are analyzed. In the rightmost column the effect of changing the parameter is summarized.

The circuit that is used for simulation is shown in Figure 3.1. The only difference between this circuit and the ones used by Palem[6] and others is that no separate noise voltage source is included in the simulations in this circuit. Noise in this circuit is generated in the simulation by the transistors according to the umc65ll N/P_12_llrvt model, while the input is noise free. The impact of having an input with noise is considered in the next chapter.

The PMOS is chosen to have double the width of the NMOS in order to compensate for difference in carrier mobility. When the PMOS and NMOS are not perfectly balanced there will be a difference in probability for a rising and falling edge. With a factor of 2 the PMOS and NMOS are approximately balanced, it is assumed to be enough to not influence the results. In future research the influence of this factor may be investigated further.

The load is chosen to be a capacitor of 10fF. This is chosen to approximately match a fanout of four inverters at 65nm. The output capacitor is part of the filter that determines the effective noise levels, therefore noise will have even more influence on smaller loads. Further research could be put in investigating the effect of the load capacitance.

The E-P curves that are shown in this chapter and the next are derived from transient

Parameter	Value	Effect of sweep
MOS type	umc65ll N/P_12_llrvt	
NMOS and PMOS length	60nm	
NMOS gate width	80nm	A larger gate width decreases effective noise and increases speed
PMOS/NMOS width ratio	2	
V_{dd}	range: 0-2V, 10mV steps	
V_{in}	Alternating between 0V and V_{dd}	
C_{out}	10fF	
Temperature	27°C	Improves V_{th} but increases noise
Noise amplification	200x	Smooths transition from 1 to 0.5 probability
Operating frequency	1GHz	Lower operating frequency shifts transition area to lower V_{dd}
Number of bits simulated	800	

Table 3.1: Default parameters for the simulations. Bold parameters are analysed and sweeped. A summary of their effect is shown.

simulations. Each point on the E-P curves is determined by sampling a transient simulation of 800 bits. The output of the inverter is sampled after each bit period. The sample is considered correct if it is higher than $\frac{V_{dd}}{2}$ and the expected value is 1, or if it is lower than $\frac{V_{dd}}{2}$ and the expected value is 0. No setup or hold times are assumed. The fraction $\frac{Correct\ samples}{Total\ samples}$ is used as probability of correctness for the corresponding point on the E-P curve. The signal source shown in the schematic switches between 0V and V_{dd}. It has a rise and fall time of 20ps.

3.2 Influence of noise

In the previous chapter the influence of feature size on effective noise levels was explained (see equation 2.1 on page 6). The effective amount of noise generated is proportional to the square root of the channel capacitance, which is determined by its feature size. In simulations the noise can be scaled by a factor. This factor multiplies all generated noise. For this circuit only the transistors generate noise, due to their channel resistance. More noise is generated during switching than in a steady state. Therefore it is not possible to link a noise scaling factor to a fixed standard deviation of the output. This is illustrated in Figure 3.2. It shows signal standard deviation at the sample moment. For higher supply voltages the signal is stable before the sample moment. At lower voltages the signal becomes slower and has not reached a steady state at the sample moment. At a noise scaling factor of 100 the steady state noise is approximately 50mV, which is only slightly less than the 60mV that is predicted for 15 years from now. Noise scaling of 200 or even 400 times would imply even smaller transistor sizes than what can be expected in 15 years. They are included to amplify the effects that are analyzed even further, but do not represent expected noise figures in the near future.

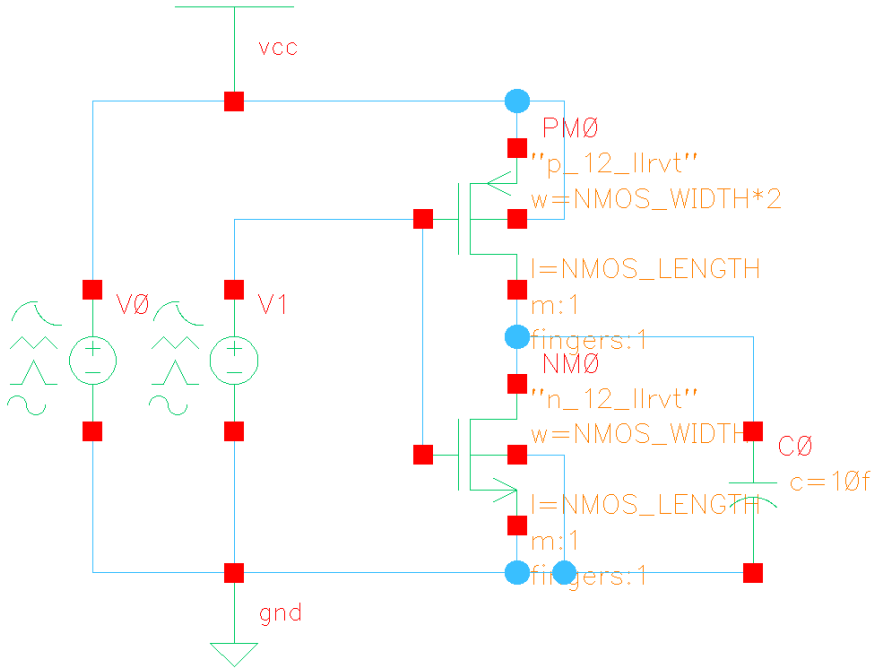


Figure 3.1: Schematic for the inverter simulations

The theoretical model for probabilistic bits by J. Kim predicts[13] that the E-P curve can be described by an error function. This model is very simple and allows for easy calculations on a system level. It assumes that the digital function is always performed without error, across the entire supply voltage range. However, channel impedance increases drastically below the threshold voltage. Even at a slightly lowered supply voltage the system will have a slower response and may produce results too late due to the increase in impedance. Therefore performance decreases at a lower supply voltage (around the threshold voltage) and the probability of correctness drops for a fixed operating frequency. The steepness of the slope depends on the amount of noise. Figure 3.3 shows this effect in simulation and also includes the E-P curve according to J. Kim[13] for a noise standard deviation of 100mV, which was previously shown to correspond to a noise scaling factor of approximately 200. These simulations confirm that the simple error function model is incorrect.

In order to make the process of noise influences more insightful, some transients were plotted. For $V_{dd} = 0.7V$ each bit period for the 1 to 0 transitions is plotted in a single plot, shown in Figure 3.4. Each of these plots is half the data gathered for a single point on an E-P curve (the other half is the 0 to 1 transitions, which are not shown). A clear trend is visible in the plots with noise scaled by a factor 50, but the trend becomes less clear when more noise is present. It can be seen that no errors are made at 50 or 100 noise scale, some at 200 and many at 400.

The situation becomes even more clear when these curves are combined into a single curve with an average and a standard deviation. This is shown in Figure 3.5. The middle line is the average, the top and bottom lines indicate the average plus or minus the standard deviation. The lines are considered to have a normal distribution, because the noise that generates the

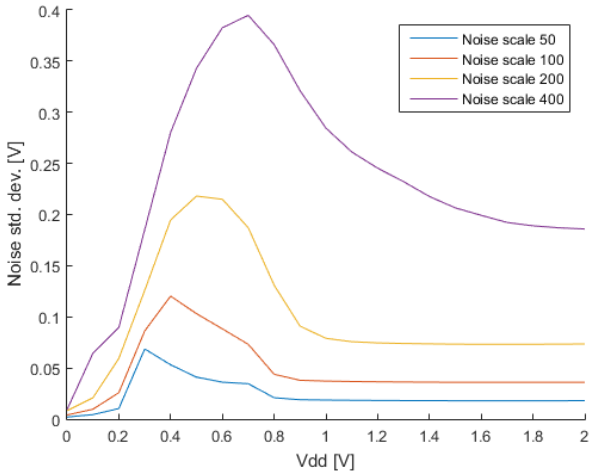


Figure 3.2: Standard deviation of the output at the sample moment.

differences is Gaussian. The first plot shows a normally functioning system with only minimal noise influence. In the second plot the average still shows normal functionality, though the asymptote seems to have moved to slightly above zero due to the noise. The third and fourth plot show that the noise further impedes the correct functioning of the system; even though the same supply and input voltage levels are used, the system does not settle to 0V before the sample moment at 1ns. The existing models of PCMOS predict that a PCMOS inverter can be described as a noise free inverter with noise superposed on the output. However, these simulations show that the propagation delay is also affected by the noise.

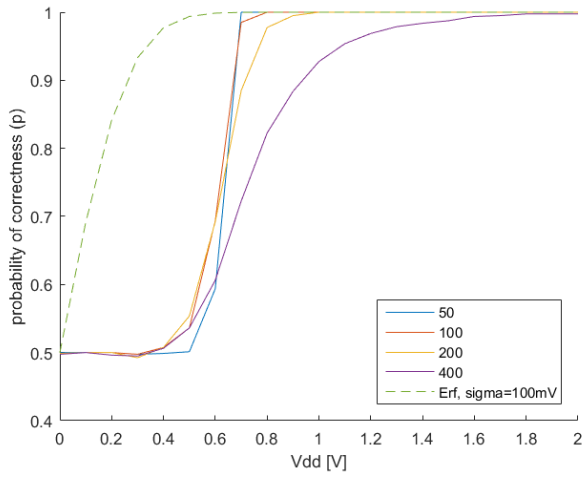
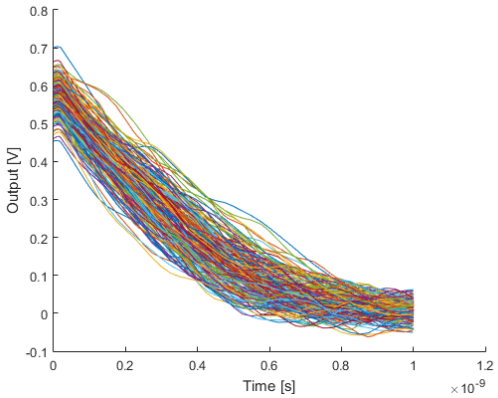
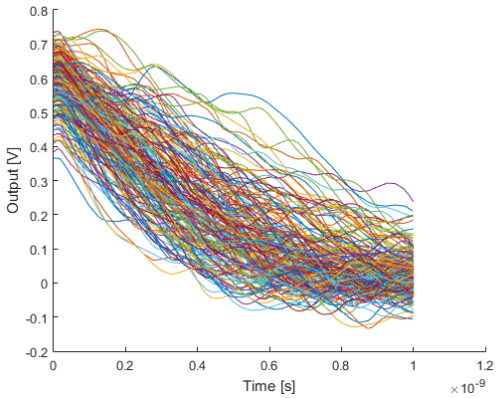


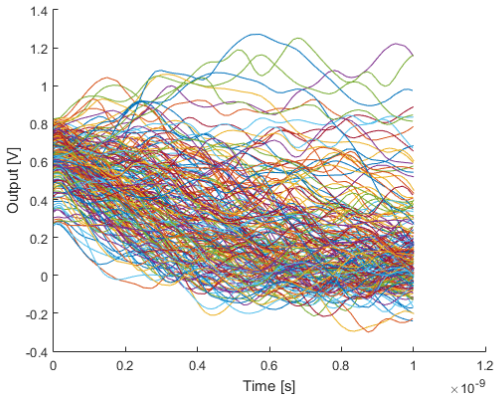
Figure 3.3: Inverter simulation with various noise scales



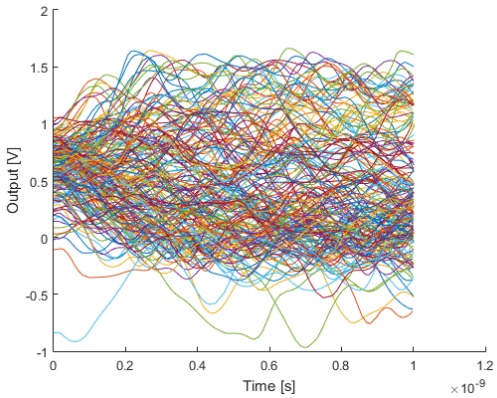
(a) Noise scale = 50



(b) Noise scale = 100

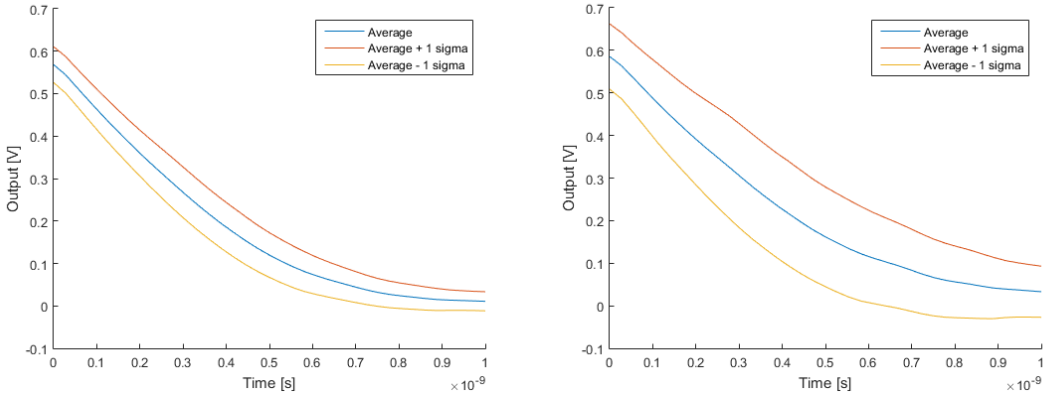


(c) Noise scale = 200



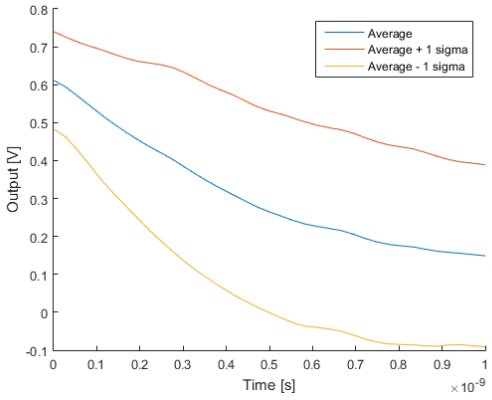
(d) Noise scale = 400

Figure 3.4: All 1 to 0 transitions for $V_{dd} = 0.7V$ combined

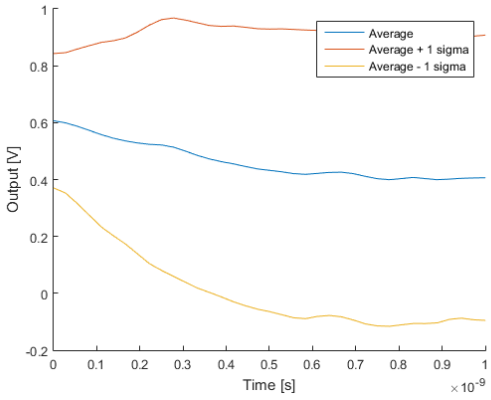


(a) Noise scale = 50

(b) Noise scale = 100



(c) Noise scale = 200



(d) Noise scale = 400

Figure 3.5: Averaged 1 to 0 transitions at $V_{dd} = 0.7V$. The middle line is the average, the top and bottom lines indicate standard deviations

3.3 Operating frequency

The inverter becomes slower when the supply voltage is lowered. It makes sense to compensate for this by decreasing the operating frequency, allowing the system more time to settle to the correct value. This method is currently used in some low power microprocessors. It is a way of keeping the system operating deterministically at voltages where it would fail at full speed.

In this research however, the goal is not to keep the performance deterministic, but to design a probabilistic system with a known fixed probability. Amongst other parameters, the supply voltage level at which the device enters its probabilistic region depends on the operating frequency. This can be seen in the drain current equation for a MOSFET. The area we want the MOSFET to operate in is sub-threshold. The drain current can then be written as[14]:

$$I_D = I_{D0} \cdot e^{\frac{V_{GS}-V_{th}}{n \cdot V_T}} \cdot (1 - e^{\frac{-V_{DS}}{V_T}}) \tag{3.1}$$

Where I_{D0} is the current that would flow at $V_{GS} = V_{th}$, V_{th} is the threshold voltage, $n = 1 + \frac{C_D}{C_{OX}}$ is the slope factor that depends on the capacitance of the depletion layer and the capacitance of the oxide layer, V_{DS} is the drain source voltage and $V_T = \frac{k_b T}{q}$ is the thermal voltage.

In the case of discharging the load capacitor, the input is stepped up until $V_{GS} = V_{dd}$. It is assumed that this step is much faster than the discharge time of the load capacitor. The voltage across the load capacitor (V_{load}) drops due to the current discharged from it by the MOSFET. V_{load} is equal to the drain source voltage of the NMOS. The term $(1 - e^{\frac{-V_{DS}}{V_T}})$ is negligible for $V_{DS} \gg V_T$. At 300K, $V_T = 25.9mV$. $V_{DS} = V_{load}$, which should be discharged until at least half the supply voltage. For any $V_{dd} > 100mV$ it holds that $V_{DS} > 2 \cdot V_T$, therefore the term is neglected. The load voltage can be written as:

$$V_{load}(t) = V_{dd} - \frac{1}{C_{load}} \int_0^t I_D dt \tag{3.2}$$

None of the terms left in I_D depend on time, therefore they can be placed in front of the integral as constants. We are in this case looking for the moment in time that V_{load} reaches $\frac{V_{dd}}{2}$, which is the transition from a binary 1 to 0 in our ideal case. In other words, we are looking for the moment when:

$$\frac{V_{dd}}{2} = V_{dd} - \frac{1}{C_{load}} I_{D0} e^{\frac{V_{dd}-V_{th}}{n V_T}} \cdot t \tag{3.3}$$

Which can be rewritten as:

$$t = \frac{V_{dd} \cdot C_{load}}{2 \cdot I_{D0} \cdot e^{\frac{V_{dd}-V_{th}}{n \cdot V_T}}} \tag{3.4}$$

The result is an equation for the propagation delay. It goes exponential with V_{dd}, until $V_{dd} \ll V_{th}$. An approximation of this curve is plotted in Figure 3.6. The parameters used are: $I_{D0} = 1\mu A, T = 300K, C_{load} = 10fF, V_{th} = 0.5V, n = 1$ ($C_{OX} \gg C_D$). In case C_{OX} is not much larger than C_D, the line will keep the exponential behaviour but be less steep; the propagation delay at low voltages will then be shorter.

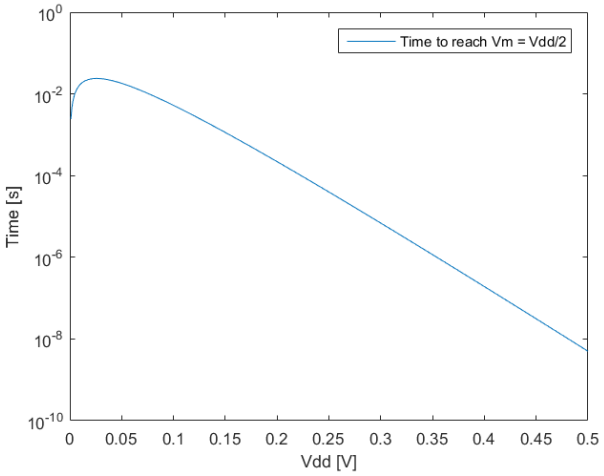


Figure 3.6: Propagation delay plotted for sub threshold operation.

The exponential behaviour of the propagation delay for different supply voltage levels is inconvenient for the speed of CMOS circuits. Lowering the supply voltage for a fixed probability requires a much lower operating frequency. In other words, the probabilistic area of operation can be shifted to lower V_{dd} by greatly lowering the operating frequency.

In this low V_{dd} area, noise is a larger problem. The errors due to noise that were described by Palem and others re-emerge. Even if the operating frequency is lowered enough to allow the signal to settle, the signal to noise ratio (SNR) may be very low and a sample may be misinterpreted. In other words, if the CMOS circuit is still fast enough at low V_{dd}, errors will be made due to noise in the output signal.

Three different operating frequencies were simulated for a part of the E-P curve. The default parameters of table 3.1 were used. Lower operating frequencies make the simulations long. Therefore only the transition area was simulated instead of the entire E-P curve. This results in a decent resolution for the relevant part of the curve. The results are shown in Figure 3.7.

The yellow, red and blue line represent parameter sweeps on the supply voltage for a specific operating frequency. At the sample moment the standard deviation of the noise was measured. The standard deviations were approximately the same. For each supply voltage, the measured standard deviation was used to calculate the expected performance due to noise according to Equation 2.3 on page 10.

The shift in the probabilistic area due to lowering the operating frequency can clearly be seen. The noise levels at the sample moment were also measured and filled into equation 2.3 on page 10. The result is plotted as the dotted line. It shows that if the operating frequency is lowered

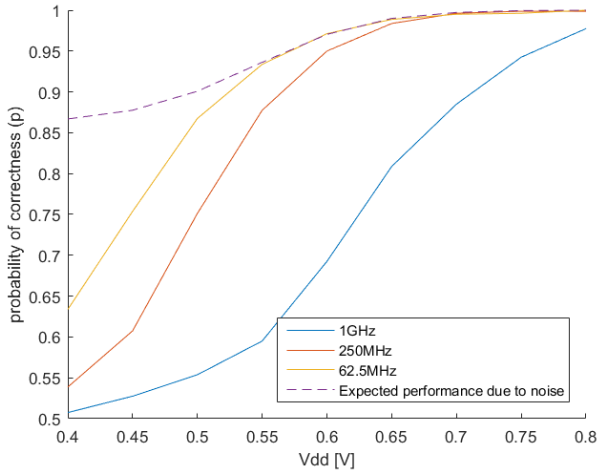


Figure 3.7: Part of the E-P curve for multiple operating frequencies. The dotted line represents the maximum performance according to equation 2.3 on page 10.

enough, noise becomes the performance limiting factor. In other words, equation 2.3 describes the maximum probability for a specific amount of noise.

3.4 Transistor size

Until now the assumption was that minimum size transistors are used. In some cases however more current may be required, for example because of a large load or when a higher speed is required. The transistors are then scaled in one dimension, the width. This increases the parasitic capacitance and decreases channel resistance, which decreases effective noise levels. The output current also increases, which is necessary to charge a larger load. Or, if the load is the same, it is charged faster. Faster charging and lower noise both increase the probability of correctness. Scaling up transistors does come at an energy price. Larger transistors have bigger gate capacitance that is charged and discharged at each state change. Cross bar current is also increased.

Scaling transistors gives a designer the possibility to create multiple PCMOS gates that operate at the same voltage, but have different probabilities of correctness when operated at the same frequency, due to the difference in propagation delay and effective noise level. Further research should be put into the applications of this, considering the increased energy consumption.

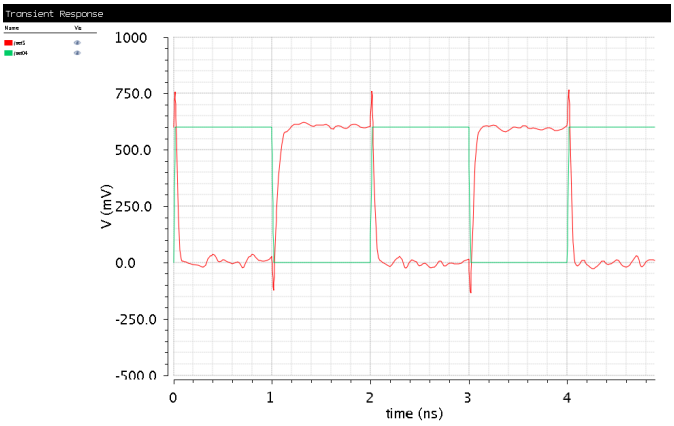
Besides decreasing the noise, larger parasitic capacitors also increase the capacitive coupling between input and output. Due to the inverting nature, the output is at V_{dd} when the input is at 0V. Therefore in steady state a voltage of V_{dd} is across the parasitic capacitors when the input is at 0V. V_{in} is then stepped up to V_{dd} in 20ps. This pushes the potential of the electrons in the parasitic capacitors up, charging the load capacitor a bit further. How much the output voltage rises due to this effect depends on the ratio of the parasitic capacitance to the load capacitance

and the steepness of the rising edge of the input voltage. After this step, the inverter has to compensate for this additional voltage, thus taking longer to discharge the load capacitor. At low supply voltages, it is not fast enough to discharge the load capacitor before the sample moment; only the input step is then visible on the output. The inverting behaviour then changes into buffering behaviour (which is always wrong). This behaviour can be seen in Figure 3.8. The effect is diminished by noise, which always drives the probability towards 0.5 due to its Gaussian nature. Therefore the effect is larger in large transistors, where noise is less dominant.

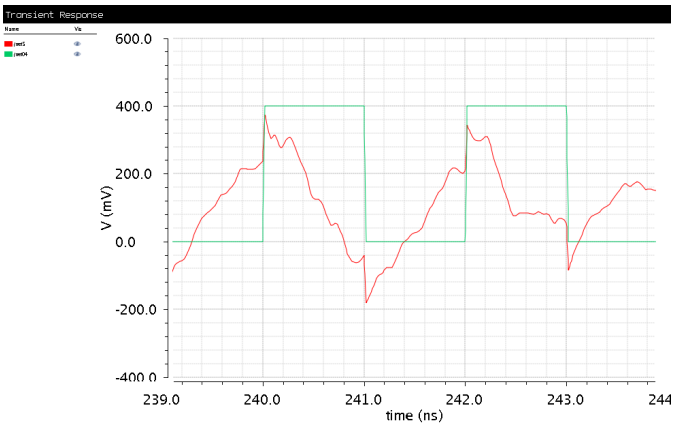
Using the default parameters, the inverter circuit was simulated for multiple transistor widths. In this simulation, the load capacitance is kept constant. The results of this simulation are shown in Figure 3.9. The three influences of increasing the transistor width that were described above can be seen in these simulation results:

1. The slope of the probabilistic area increases for increased width due to decreasing noise levels.
2. The probabilistic area shifts to lower V_{dd} for increased width due to larger currents (same effect as lowering the operating frequency).
3. The probability of the inverter drops below 0.5 due to the capacitive coupling between input and output. The effect is more clearly present for wider transistors due to lower noise levels.

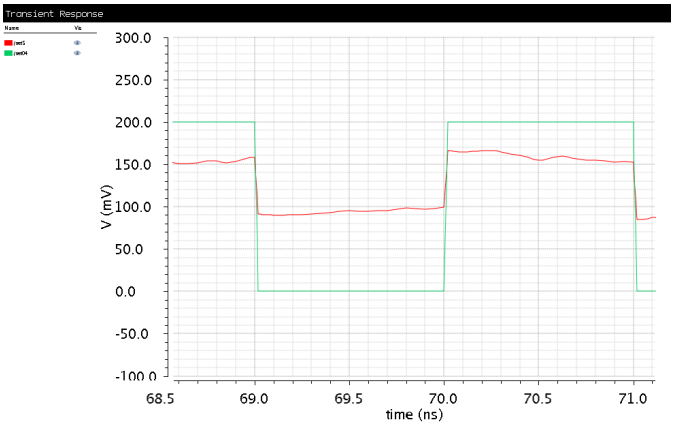
In practice a probability of less than 0.5 does not pose a problem. As soon as the probability is allowed to approach 0.5, the calculation should not be performed at all and replaced by a wire to either ground or V_{dd}, which does not use any energy and has a probability of 0.5 (given a uniform distribution between ones and zeroes). Another option is to substitute the digital function for a simpler one that provides a correct output in the majority but not all of the cases. This is called approximate computing[15].



(a) Part of the waveform at $V_{dd} = 0.6V$, behaviour is correct.



(b) Part of the waveform at $V_{dd} = 0.4V$, behaviour is mostly correct.



(c) Part of the waveform at $V_{dd} = 0.2V$, behaviour is mostly wrong.

Figure 3.8: Transients of 80u wide transistor at 3 different voltages. The green line is the input of the inverter, the red line the output.

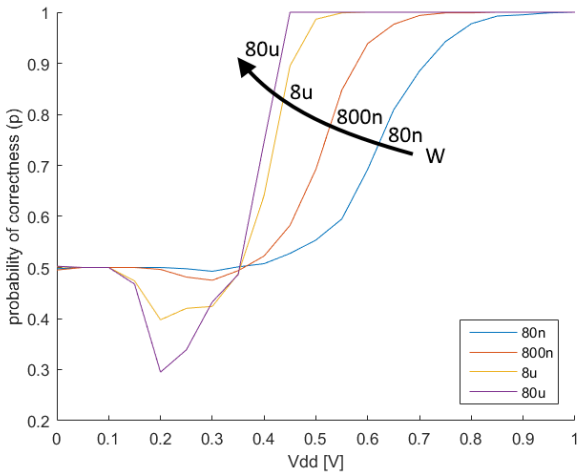


Figure 3.9: E-P curves for multiple gate widths.

3.5 Temperature

Temperature has an effect on most aspects of MOSFET performance. Increasing temperature decreases electron and hole mobility. It also increases the thermal voltage: $V_T = \frac{k_b T}{q}$. In equation 3.1 on page 21 it was already shown that an increase of the thermal voltage decreases the drain current in sub-threshold operation.

The effective noise voltage is given as $V_n = \sqrt{\frac{k_b T}{C}}$ in chapter 2. While mobility and thermal voltage influence the speed of the transistor and therefore shift the probabilistic area, the temperature also influences noise and therefore an increase in temperature widens the probabilistic area.

The three effects mentioned above all have a negative influence on probability of correctness. One parameter that is improved by increasing the temperature is the threshold voltage. In a paper from 1991 Z. Prijic[16] showed that the threshold voltage is significantly lowered at high temperatures. This keeps the transistor out of the sub-threshold region longer, increasing the performance. This effect is shown for (old and large) Si-gate CMOS transistors in Figure 3.10.

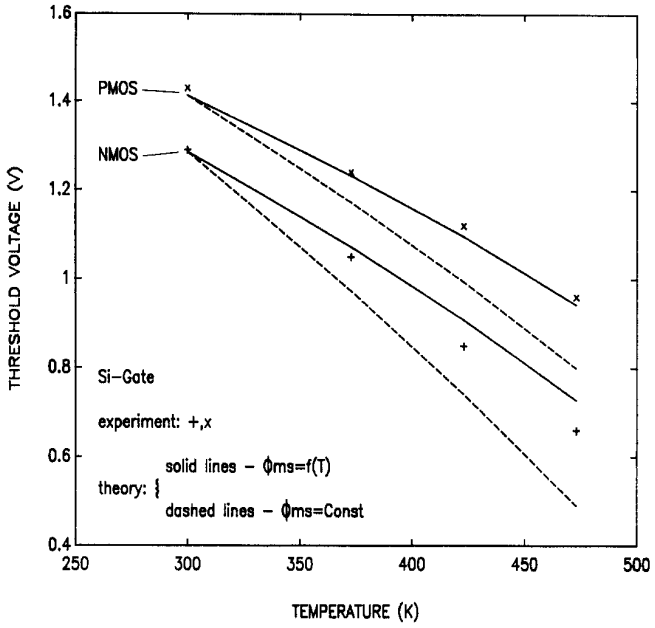


Figure 3.10: Threshold voltage temperature dependency for Si-gate CMOS transistors, as shown in [16].

The effect of the lowered threshold voltage on probability was simulated for five different temperatures using a parameter sweep. The default parameters were used. Note that the simulation does not increase the thermal noise at higher temperature. It is a recommendation that this simulation is repeated with the inclusion of increased thermal noise. The results are shown in Figure 3.11. It shows that increasing the temperature improves probability around the threshold voltage, while it has little effect at high V_{dd}.

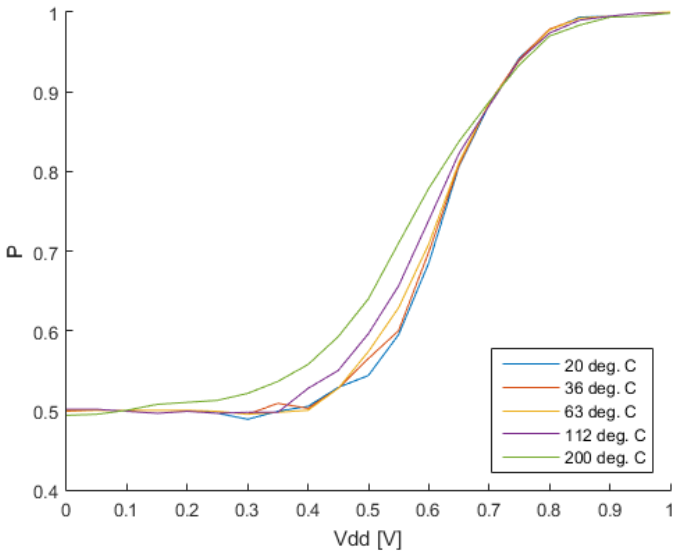


Figure 3.11: E-P curves for multiple temperatures (unit: degrees Celsius). Thermal noise is not increased with temperature.

3.6 Conclusion on PCMOS inverter performance

The probability of correctness of a PCMOS inverter can be greatly influenced by the parameters that were analyzed. Existing models of PCMOS are shown to be incomplete. The E-P curves that are predicted by existing models are shown to be correct only in cases when the output signal is allowed enough time to settle before the sampling moment. The time required increases exponentially when lowering V_{dd} in the sub-threshold region. The operating frequency of a PCMOS inverter can be lowered in order to shift the probabilistic region of the E-P curve to lower V_{dd}. Increasing the width of a transistor both increases the speed and decreases the noise. Increasing the transistors' speed shifts the probabilistic region to lower V_{dd} (similar to lowering the operating frequency). Less noise results in a shorter probabilistic region (the transition from 0.5 to 1 probability becomes steeper). Higher temperature also increases noise, but at the same time it lowers the threshold voltage.

4 Using probabilistic hardware in a design

In the previous chapters, only a single gate PCMOS circuit was considered. In this chapter, a series of PCMOS building blocks is considered. The goal of this chapter is to provide insight into the influences that linked probabilistic building blocks have on each other when connected. This goal is related to the ultimate goal for probabilistic systems: Being able to easily design a probabilistic system with a known expected error that uses the least amount of energy possible. For this we need:

1. An effective way to determine the expected error at the output of a probabilistic system for a specific energy distribution.
2. An algorithm to find the optimal energy distribution for a specific expected error.

Some steps are already taken towards algorithms for finding optimal energy distributions. A notable example is a series of papers by M. Lau[12][11] in which he designs a probabilistic ripple carry adder and multiplier. His methods are purely mathematical and use deterministic full adders with output coupled noise sources to model probabilistic full adders. However, it was shown in the previous chapter that this model does not hold for low supply voltages. Another example is a paper by J. George[17] in which he uses a method called *biased voltage scaling* to create an energy distribution for a ripple carry adder. Biased voltage scaling means assigning supply voltages to specific parts of a system (in this case a ripple carry adder) according to a predefined function that is optimal with respect to energy consumption for that configuration. A ripple carry adder was chosen because it has a regular structure and is commonly used. The validation of his method uses deterministic logic blocks with output coupled noise sources, according to existing models. Therefore it ignores the influences that each full adder has on its successor in the chain. This method also does not guarantee finding the optimal energy distribution, because it is based on approximate models and has no iterative feedback to correct for approximations.

All of the existing methods and models lack a proper way to accurately determine the expected error at the output of a probabilistic system. Therefore in the first part of this chapter the influences of a block on subsequent blocks is analyzed. A ripple carry adder is used as an example. In the second part of the chapter, the problem of finding an optimal energy distribution is re-examined using the newly gained insights.

4.1 Finding the expected error of a probabilistic system

The first step towards being able to design a probabilistic system with a known expected error for the lowest possible energy consumption is creating a way to predict the expected error for a given design. Using physical simulations to determine the behaviour of the entire system may

take a long time depending on the size. A method for predicting system level behaviour based on the behaviour of the individual blocks is desired. Traditional digital CMOS circuits solve this by characterizing the gates through parameters such as propagation delay and rise/fall time of signals. Through simulation using these simplified parameters it is determined whether the outputs are calculated in time or not.

For probabilistic systems we would like the same situation. Characterize each basic building block only once, then do a relatively simple simulation in order to estimate performance. Thus, information is needed on how connected probabilistic systems influence each other and the output.

Similar to traditional digital logic, signals propagate through the system at a certain speed. Unlike traditional digital logic however, there is a large uncertainty in the propagation of the signals. Voltage noise can cause misinterpretations at the sample moment, but during transitions voltage noise also causes an uncertainty in the propagation delay. This is illustrated in Figure 4.1. When the curve around the halfway point is approximated by a straight line, the probability density function (PDF) in the time dimension is a scaled version of the PDF in signal dimension, which is a normal distribution.

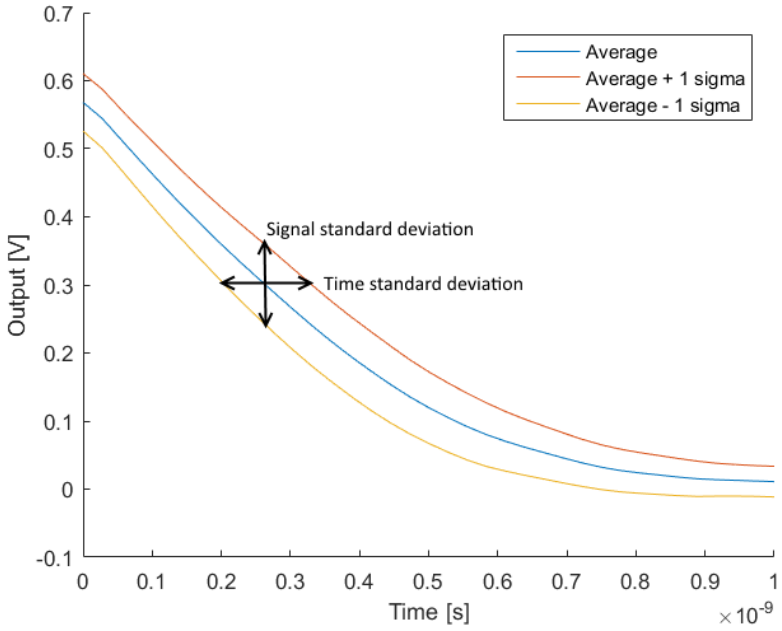


Figure 4.1: Voltage noise during transitions causes time noise.

When two of these probabilistic systems are connected, the situation becomes slightly more complicated. Consider two probabilistic systems that both have a known PDF of their propagation delay, that can be described by a normal distribution. The parameters used in this example are $\mu_1 = 10$ns, $\sigma_1 = 1$ns, $\mu_2 = 15$ns and $\sigma_2 = 1$ns. The starting time of the second system has a PDF that is equal to the propagation delay PDF of the first system. Its propagation delay is also a normal distribution, that is shifted in time depending on the actual starting time. Therefore the PDF of the propagation delay of the second system can be calculated as the convolution of the PDF of the propagation delay of the first system and its own propagation delay PDF

assuming a starting time of $t = 0$. The convolution of two Gaussian functions can be calculated easily using a Fourier transform, it results[18] in another Gaussian function, with properties $\mu = \mu_1 + \mu_2$ and $\sigma = \sqrt{\sigma_1^2 + \sigma_2^2}$. This is visualized for the two example systems in Figure 4.2. The standard deviation of combined probabilistic systems goes as $\sqrt{N}$, where N is the number of systems combined.

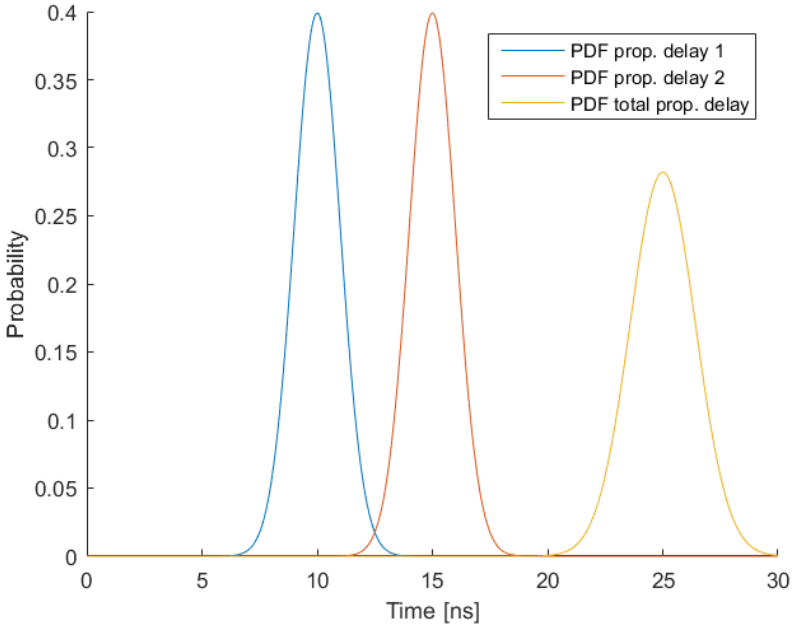


Figure 4.2: The convolution of two Gaussian functions results in another Gaussian function width a larger standard deviation.

4.1.1 Ripple carry adder

The interdependence of connected probabilistic systems is analysed for the specific case of a ripple carry adder. It is a commonly used digital building block that performs the mathematical addition of two binary numbers. There are many different ways to implement such an addition in hardware, but one of the most simple implementations is the ripple carry adder. A block diagram of a four bit ripple carry adder is shown in Figure 4.3.

The ripple carry adder consists of multiple identical blocks: One bit full adders. Each full adder has two data inputs and a carry input. It produces a sum and a carry output. The carry output is used as an input for the next block. The output of stage n can therefore only be calculated after stage n-1 has produced its carry output. This effect makes the ripple carry adder ideal for analyzing the influence of blocks on each other: All the blocks are identical, but one of their inputs is determined by the previous block.

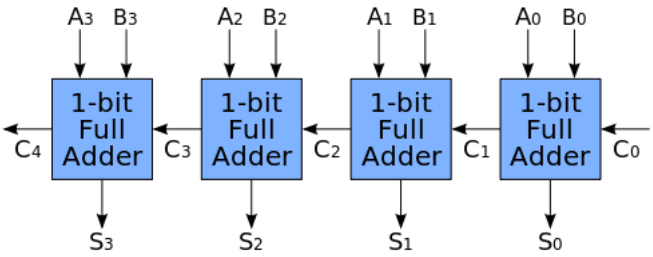


Figure 4.3: Block diagram of a 4 bit ripple carry adder. Source: commons.wikimedia.org.

A	B	Cin	Sum	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Table 4.1: Truth table of a full adder.

Full Adder simulations

A full adder is a digital block that performs a single bit addition. Its truth table is shown in Table 4.1. This function can be implemented in many different ways. A common implementation using standard logic gates is shown in Figure 4.4.

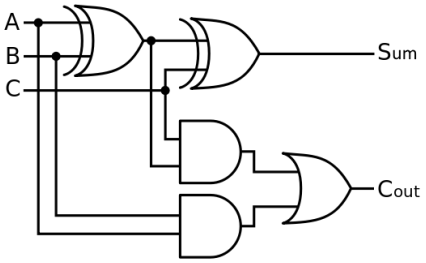


Figure 4.4: Block diagram of a full adder. Source: commons.wikimedia.org.

When implementing a full adder in CMOS, this circuit can be optimized to a less readable version that uses less transistors. It is shown in Figure 4.5. In this circuit, two intermediate steps can be distinguished. The inputs A, B and C are used to calculate $\overline{\text{Cout}}$. Its value is then used to calculate both Cout and $\overline{\text{Sum}}$ (also using the inputs A, B and C). Finally, Sum is calculated by inverting $\overline{\text{Sum}}$. Therefore Cout can be viewed as a PBit that is the output of a series of 2 probabilistic circuits and Sum as a PBit that is the output of a series of 3 probabilistic circuits.

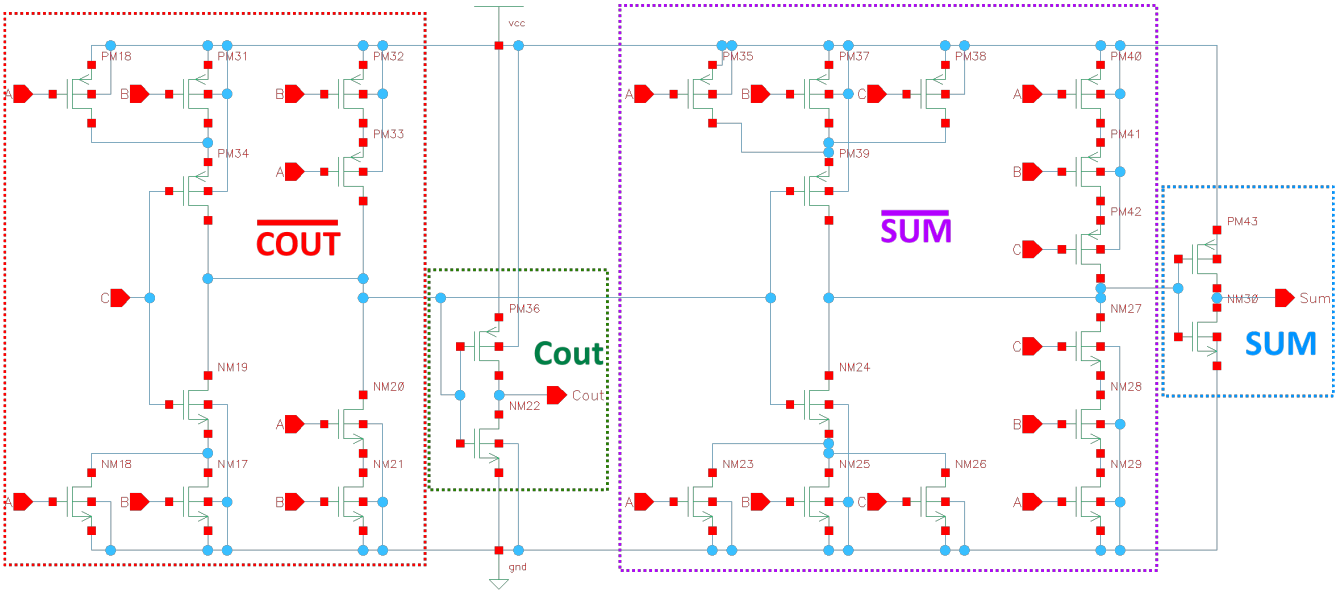


Figure 4.5: Full adder circuit implementation.

Comparing Sum and Cout to a PCMOS inverter, Sum needs the lowest clock frequency, followed by Cout. Simulations were performed to confirm this behaviour. A full adder was simulated using the default parameters of chapter 3. The Cout and Sum outputs were simulated using a parameter sweep on the supply voltage. The resulting E-p curves are shown in Figure 4.6.

Even though a full adder is a much more complex circuit than an inverter, all the rules from the previous chapter are assumed to still apply to it, because it is a series connection of multiple CMOS sub-circuits. A property of a digital CMOS circuit is that for every possible combination of logical inputs its load is always connected to either ground or V_{dd}. Therefore the more complex CMOS circuits that calculate $\overline{\text{Cout}}$ and $\overline{\text{Sum}}$ are similar to a CMOS inverter, but with a more complex channel that has multiple gates. Figure 4.7 shows a simulation of a full adder at multiple noise levels, which confirms that it behaves similar to an inverter.

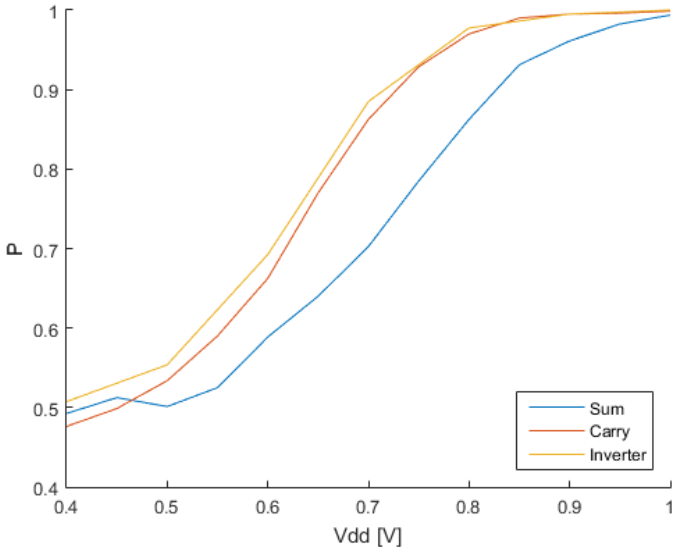


Figure 4.6: Comparison of the E-P curves of an inverter and a full adder for a noise factor of 200.

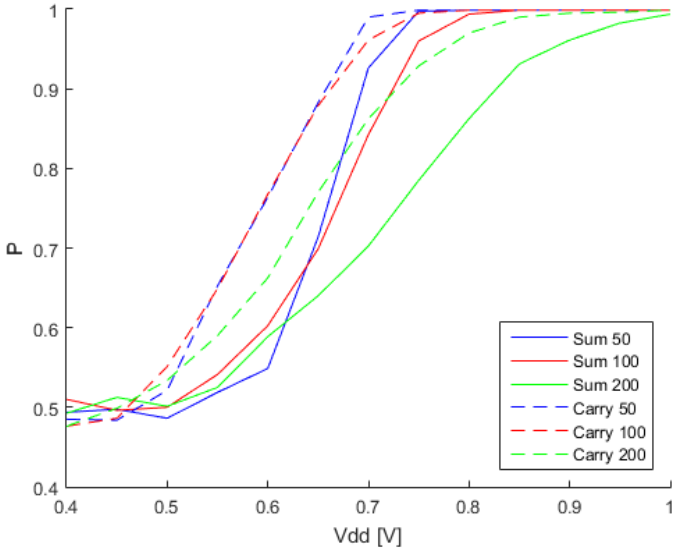


Figure 4.7: E-P curve simulations of the full adder circuit for several noise factors.

Propagation of errors

Since each part of a large probabilistic system is a probabilistic system itself, errors can be made in any part. These errors then have a probability of propagating to the output depending on the path to that output. Any subsequent part that uses an erroneous signal as an input can either produce a wrong output itself (propagating the error) or a correct output (stopping the propagation). In other words, the probability of correctness of the sum output of stage n of the ripple carry adder depends on the probability that its inputs are correct, but also on the probability of the block itself. If the correct calculation is done on wrong data, the output may still be incorrect. From the truth table (Table 4.1) it can be seen that changing only the carry input will always change the sum output. Therefore a wrong carry input results in a wrong sum output, unless another error is made. A correct sum output is thus produced when both the input and the calculation are correct, or both are wrong:

$$P(s'_{i+1} = s_{i+1}) = p^s_{i+1} * p^c_i + (1 - p^s_{i+1}) * (1 - p^c_i) \tag{4.1}$$

Where s'_{i+1} is the correct sum output of stage i+1, s_{i+1} is the actual output of stage i+1, p^s_i is the probability of calculating the correct sum output at stage i given correct inputs and p^c_i is the probability of calculating the correct carry output at stage i. In this expression the carry output of stage i-1 is equal to the carry input of the next stage. For identical full adders placed in the same environment, it is assumed that $\forall i : p^c_i = p^c$ and $\forall i : p^s_i = p^s = 1$. If the probability of the carry output is 1, the expression simplifies to $P(s'_{i+1} = s_{i+1}) = p^s_{i+1}$. If the probability of the carry output is 0.5 (randomly chosen value), the expression simplifies to $P(s'_{i+1} = s_{i+1}) = 0.5 * (p^s_{i+1} + (1 - p^s_{i+1})) = 0.5$.

Following this way of calculation, the probability of correctness for each sum and carry output can be calculated. M. Lau did this in a previously mentioned paper[11] and found the following expression for the probability of an error in the sum output of a full adder stage in a ripple carry adder:

$$P(s'_{i+1} = \neg s_{i+1}) = \frac{1}{2} - (\frac{1}{2} - p^s_{err,i+1}) \cdot [\prod_{j=1}^{i} (\frac{1}{2} - p^c_{err,j}) + \sum_{k=1}^{i} \prod_{l=k}^{i} (\frac{1}{2} - p^c_{err,l})] \tag{4.2}$$

Note that in this expression different conventions of notation are used than in this report. $p^c_{err,i}$ denotes the probability of a calculation error in the carry of stage i. $p^s_{err,i}$ denotes the probability of a calculation error for the sum in stage i. In this report the probability of correctness is used rather than probability of error. Rewriting expression 4.2 results in:

$$P(s'_{i+1} = s_{i+1}) = \frac{1}{2} + (p^s_{i+1} - \frac{1}{2}) \cdot [\prod_{j=1}^{i} (p^c_j - \frac{1}{2}) + \sum_{k=1}^{i} \prod_{l=k}^{i} (p^c_l - \frac{1}{2})] \tag{4.3}$$

If all full adders are identical, $\forall i : p_i^s = p^s$ and $\forall i : p_i^c = p^c$. The equation then simplifies to:

$$P(s_{i+1}^{'} = s_{i+1}) = \frac{1}{2} + (p^s - \frac{1}{2}) \cdot [(p^c - \frac{1}{2})^i + \sum_{k=1}^{i} (p^c - \frac{1}{2})^k] \tag{4.4}$$

In the hypothetical situation that the sum is always calculated correctly given correct inputs ($p^s = 1$), the probability of the sum output only depends on the probability of the carry output of the previous stage (this can be seen by filling in $p_i^s = 1$ in equation 4.1). Using $p^s = 1$ equation 4.4 results in:

$$P(s_{i+1}^{'} = s_{i+1}) = \frac{1}{2} + \frac{1}{2} \cdot [(p^c - \frac{1}{2})^i + \sum_{k=1}^{i} (p^c - \frac{1}{2})^k] \tag{4.5}$$

Consider an infinite series of full adders. The last full adder then has the accumulated carry probability of an infinite amount of carry calculations. For $\lim_{i \to \infty}$ this expression is solved as:

$$\lim_{i \to \infty} P(s_{i+1}^{'} = s_{i+1}) = \frac{1}{2} + \frac{1}{2} \sum_{k=1}^{\infty} (p^c - \frac{1}{2})^k = \frac{1}{2} + \frac{1}{2} \cdot \frac{p^c - 0.5}{1 - (p^c - 0.5)} \tag{4.6}$$

This hypothetical limit shows that the influence of an error in the carry diminishes (if not, the accumulated error of an infinite amount of carries would have fixed the probability at 0.5). In other words, an error in the carry is not expected to propagate far up the chain towards more significant bits.

4.1.2 Ripple carry adder simulations

In section 4.1.1 a full adder was analysed. It is viewed as a sequence of probabilistic sub-systems, 2 in series for calculating the carry and 3 in series for calculating the sum. A ripple carry adder (RCA) is a sequence of full adders. Therefore the carry output of stage n of the RCA is a connection of $2n$ probabilistic sub-systems in series and the sum output of stage n is a connection of $2n + 1$ probabilistic sub-systems in series. Therefore both the sum and carry output of each stage have a lower probability of correctness than the previous stage (given all parameters are the same). In Cadence IC four full adders were connected to form a four bit ripple carry adder. It was simulated with the default parameters, except for the noise amplification, which is set to 100. During the simulations each full adder is given the same supply voltage. This is done in order to remove the effect of differences in binary interpretations of voltages. The results are shown in Figure 4.8. Sum n and Carry n represent the sum and carry output of the nth stage of the ripple carry adder respectively, counting starts at the LSBit.

According to Lau, the sum outputs 2,3 and 4 can also be predicted using equation 4.4. This (simplified) version can be used here because each full adder is identical and has the same supply voltage. The resulting E-P curves of each stage then always have the same shape as the first

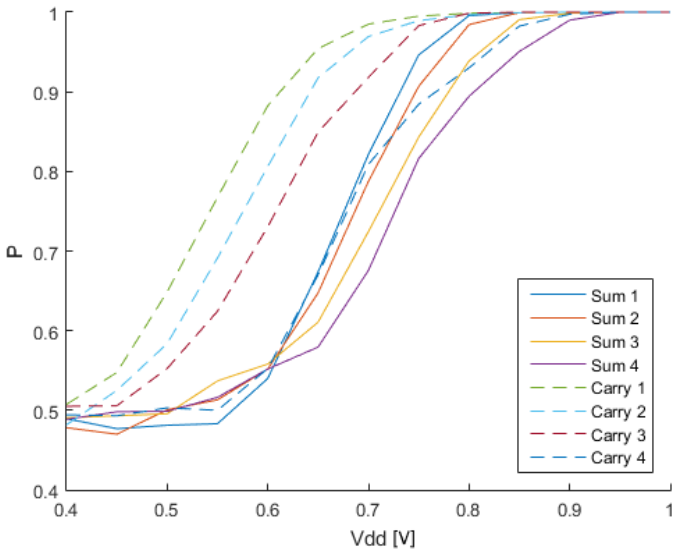


Figure 4.8: E-P curves for a four bit ripple carry adder with noise scaled 100 times.

stage sum and carry outputs, because they are always a linear combination of the E-p curves of the first stage. It has been shown however, that each stage has a probabilistic area that is *shifted* with respect to the previous stage, not scaled. Moreover, equation 4.4 predicts that each sum output has the same probability when the carry probability is equal to 1. Figure 4.9 shows the simulated sum E-P curves next to their counterparts calculated using equation 4.4. Figure 4.10 shows the same for the carry outputs. Both predictions are inaccurate because they ignore the effect of propagation delay.

4.1.3 Conclusion on ripple carry analysis

The existing model of a probabilistic ripple carry adder works on the same assumption as the existing models of a single component probabilistic system, namely that errors are only caused by voltage noise and not by errors due to missed deadlines. Models for propagation of errors are developed, but do not provide accurate predictions of probability due to this assumption. Therefore better models are required that take propagation delay and missed deadlines into account.

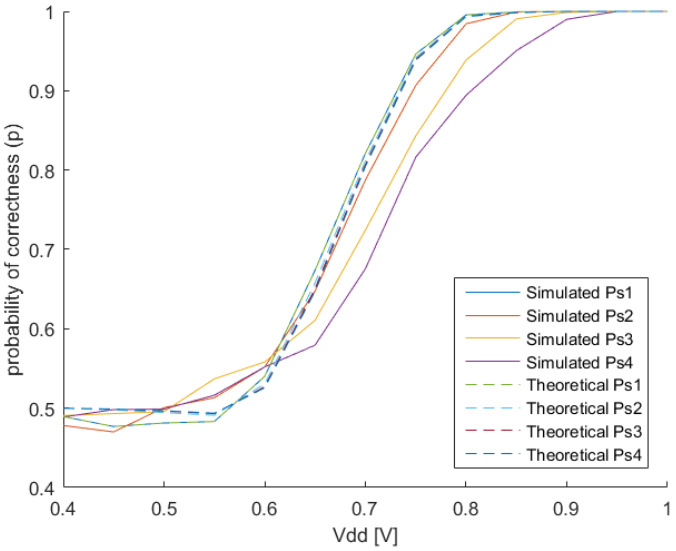


Figure 4.9: Calculation of sum errors using the model by M. Lau.

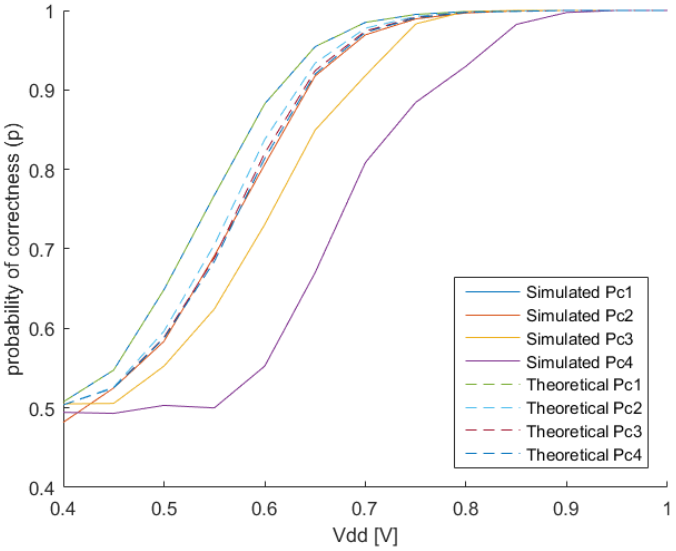


Figure 4.10: Calculation of carry errors using the model by M. Lau.

4.2 Finding the optimal energy distribution

In the previous section the interaction of probabilistic building blocks in a system was analyzed. In this section that knowledge is used to better describe the problem of finding the optimal energy distribution for a given total error. Until now, the total error has not been used as a measure of performance, but rather the probability of correctness. Therefore definitions of the total error are first introduced.

Finding the optimal energy distribution for a given total error is a complicated problem, because there are many degrees of freedom. Especially since it has become clear from the ripple carry adder analysis that blocks have a substantial influence on subsequent blocks. Therefore it is important to first define the problem mathematically and analyze it.

4.2.1 Definition of error and optimum

Consider a system that consists of N digital blocks with M outputs. A block is considered to have a known probability of correctness, which can depend on both its assigned energy E and the assigned energy of previous blocks (due to propagation delay). An error produced by a block has a probability of propagating to each of the outputs, p_m^{prop}. If it is not connected to an output m at all, $p_m^{prop} = 0$. Each of the outputs has an assigned weighing factor W_m that accounts for the impact of that output(e.g. 8 for the fourth bit of a ripple carry adder). The total error produced by a block i is then:

$$\epsilon_i = P_i(E_1, E2, .., E_N) \sum_{j=1}^{M} p_m^{prop} \cdot W_m \tag{4.7}$$

The total expected error is then defined as:

$$\epsilon_{total} = \sum_{i=1}^{N} \epsilon_i = \sum_{i=1}^{N} P_i(E_1, E2, .., E_N) \sum_{j=1}^{M} p_m^{prop} \cdot W_m \tag{4.8}$$

For an isolated block, ϵ_i will be equal to its E-P curve multiplied by a weighing factor. If more blocks are connected to the output of a block however, their performance may also depend on the performance of their predecessor. Both ϵ_i and E_i are bounded by the half open interval $[0, \infty)$. The physical implications of these bounds are:

- A block can only use energy, not provide it ($E \geq 0$).
- A block can only cause errors, not correct errors made by other parts of the system ($\epsilon \geq 0$).
- The amount of energy consumed by the block can be arbitrarily large.
- The error contribution of a block can be arbitrarily large.

In practice the upper bound of the energy consumption is not realistic, because the device will break down at a certain voltage. One could compensate for this by applying an artificial boundary: Make the error contribution of the block equal to infinity for any energy higher than the desired energy boundary. No optimal solution will then incorporate that voltage. The arbitrarily large error contribution can be realized by assigning infinity as the weighing factor of an output. This forces any viable solution to ensure that the probability of correctness for that output is 1.

From the ripple carry adder analysis it became clear that changing the energy of one block may influence the probability of correctness of another block. When the energy assignment of a single block is changed, the difference in the total error depends on the difference in error of each block:

$$\frac{\mathrm{d}\epsilon_{total}}{\mathrm{d}E_i} = \sum_{j=1}^{N} \frac{\partial f_j}{\partial E_i} \tag{4.9}$$

If a block j is not influenced by block i, its partial derivative will be zero. If a block j *is* influenced by the change in energy assigned to block i, this derivative is assumed to be smaller than or equal to zero. In other words: If more energy is assigned to a block i, every block is assumed to make the same amount of errors or less. Block i finishes its calculations sooner, giving the blocks that use its output more time to calculate their answer. This effect is similar to the bit period simulation of the previous chapter (section 3.3), where it was shown that the performance of a probabilistic inverter increased when the bit period was increased. The consequence of this effect is that the derivative of total error to energy assigned is always smaller than or equal to zero:

$$\frac{\mathrm{d}\epsilon_{total}}{\mathrm{d}E_i} \leq 0 \tag{4.10}$$

Going back to the original problem, we want to find the optimal energy distribution for a given total error. In this case the optimal energy distribution is the one that uses the least amount of energy. The total energy consumption is equal to the sum of the energy consumption of each block:

$$E_{total} = \sum_{i=1}^{N} E_i \tag{4.11}$$

The optimal distribution is that which minimizes equation 4.11 while keeping its total error according to equation 4.8 within the given limit. A smaller total error will in most cases also suffice, but in practice the total error will be exactly equal to the maximum allowed error for minimal energy consumption, because if it is less, some additional energy can be saved until the maximum allowed error is reached.

4.2.2 Algorithm for finding the optimum

As an example of a probabilistic system, a system with two elements is considered, in this case a two bit binary AND operation. The two elements are independent. Their E-P curves can be described by the same function:

$$P = 0.5 + 0.5 \cdot \text{erf}(E)^2 \tag{4.12}$$

Both propagate to one output with a probability of 1. However, element 2 has a weighing factor of 2, whereas element 1 has a weighing factor of 1. Using equation 4.7, the error contributions are found:

$$\epsilon_1 = 0.5 - 0.5 \cdot \text{erf}(E1)^2$$
$$\epsilon_2 = 2 \cdot (0.5 - 0.5 \cdot \text{erf}(E2)^2)$$

When filled into equation 4.8 it becomes:

$$\epsilon_{total} = 0.5 - 0.5 \cdot \text{erf}(E1)^2 + 2 \cdot (0.5 - 0.5 \cdot \text{erf}(E2)^2) \tag{4.13}$$

This equation can be depicted as a surface, for which the total error is equal to the height. The contour lines of this surface represent all possible combinations of E1 and E2 assignments that lead to the same total error. The surface for this system is shown in Figure 4.11, along with the contour line that belongs to a total error of 0.3. The red marker on the line represents the optimal energy distribution for that contour.

Finding the optimal energy distribution is represented here as finding the location of the red marker on this surface. Assume that we walk across this surface starting on the point E1=E2=2 and that only straight steps in the direction of E1 or E2 can be taken. A step taken then corresponds to a unit of energy less consumed. The point of least energy consumption for the desired error is the one that takes the most steps to reach for a certain height while only going up.

For such a surface the slope corresponds with the increase in error per unit decrease of energy consumption. We therefore always want to take a step int the direction that has the largest (least negative) slope. The change in slope (second derivative) indicates whether decreasing the same energy assignment the next time becomes more or less favorable. The first and second order derivatives of equation 4.12 are:

$$\frac{d}{dE}[0.5 - 0.5erf(E)^2] = \frac{-2}{\sqrt{\pi}} e^{-E^2} \text{erf}(E) \tag{4.14}$$

$$\frac{d^2}{dE^2}[0.5 - 0.5erf(E)^2] = (\frac{4}{\sqrt{\pi}} e^{-E^2} x \ \text{erf}(E) - \frac{4}{\pi} e^{-2E^2}) \tag{4.15}$$

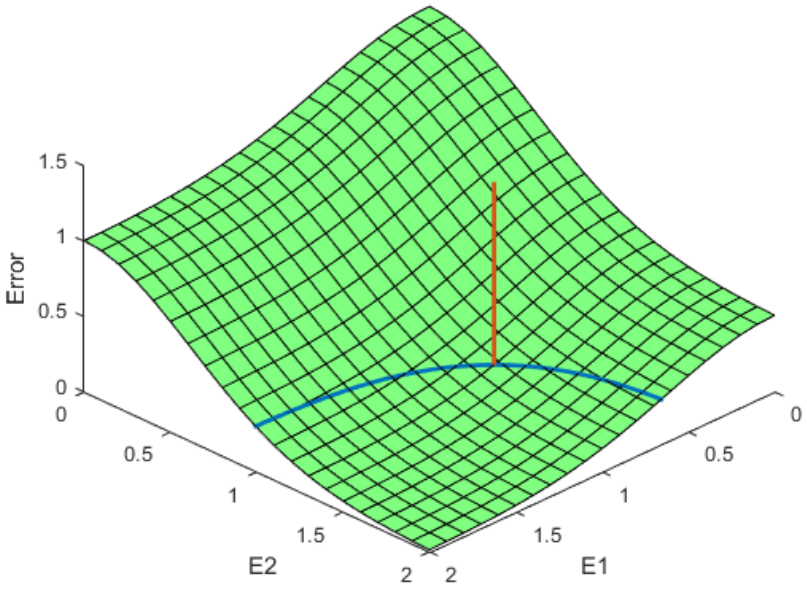


Figure 4.11: A surface formed by equation 4.13, with a contour line and optimal energy distribution for that contour line indicated

When stepping from the point $E2 = E1 = 2$ towards the origin (the highest point of the surface), an area can be defined where the first order derivative is negative and the second order derivative is positive in both the E1 and E2 direction. From equation 4.10 it follows that this surface has a negative first order derivative for any point in any direction. The second order derivative is given in equation 4.15, from this equation it can be found that both second order derivatives are positive where E2 and E1 are both larger than 0.62. The increase in expected error can be found from the step size and the derivative:

$$\Delta\epsilon_{total} = \sum_{i=0}^{N} \Delta E_i \cdot \frac{\partial f_i}{\partial E_i} \tag{4.16}$$

Since both the step size and the first order derivative are negative, the error increases for each step towards the origin. After the step the first order derivative has also changed:

$$\Delta\frac{\partial f_i}{\partial E_i} = \Delta E_i \cdot \frac{\partial^2 f_i}{\partial E_i^2} \tag{4.17}$$

which is negative while $E_i > 0.62$. If in that area a step is taken towards the origin in the direction with the largest (least negative) first order derivative, that derivative decreases (becomes more negative), making it less favorable. After a few steps the situation is reached where both first order derivatives are equal, and the most favorable step alternates between both

blocks. A sort of trench is shaped that can be followed, always ensuring the most steps are taken to reach a certain height. This is shown in Figure 4.12. Though this explanation makes it plausible that the optimal energy distribution can be found by continuously choosing the lowest first order derivative, a formal proof has not been found and is therefore not given. The algorithm followed is:

```
Double E1 = E_MAX;
Double E2 = E_MAX;
Double error = f(E1, E2);
Double step = 0.1;

while(error < MAX_ERROR)
{
  //Check if we are at a border
  if(E1 == 0)
  {
    E2 -= step;
  }
  else if(E2 == 0)
  {
    E1 -= step;
  }
  //Take a step in the direction that gives the
  //lowest decrease in performance
  else if(f(E1 - step, E2) < f(E1, E2 - step))
  {
    E1 -= step;
  }
  else
  {
    E2 -= step;
  }
  //Update the current error
  error = f(E1, E2);
}
```

In this algorithm, f is the function that calculates ϵ_{total} according to equation 4.8. The algorithm is defined for two dimensions, but can be extended to more dimensions.

Even though this algorithm takes rather large steps, it approximates the optimal energy point nicely. To show this in more detail, the optimal energy point for each height is plotted along with the curve that the algorithm follows in Figure 4.13 as a line of red markers.

When the algorithm reaches the point where the second order derivative becomes negative, something curious happens. A discontiuïty appears in the optimal path. Because the algorithm takes incremental steps, it cannot follow the discontinuïty. This proves that the algorithm cannot find the optimal energy distribution in every situation.

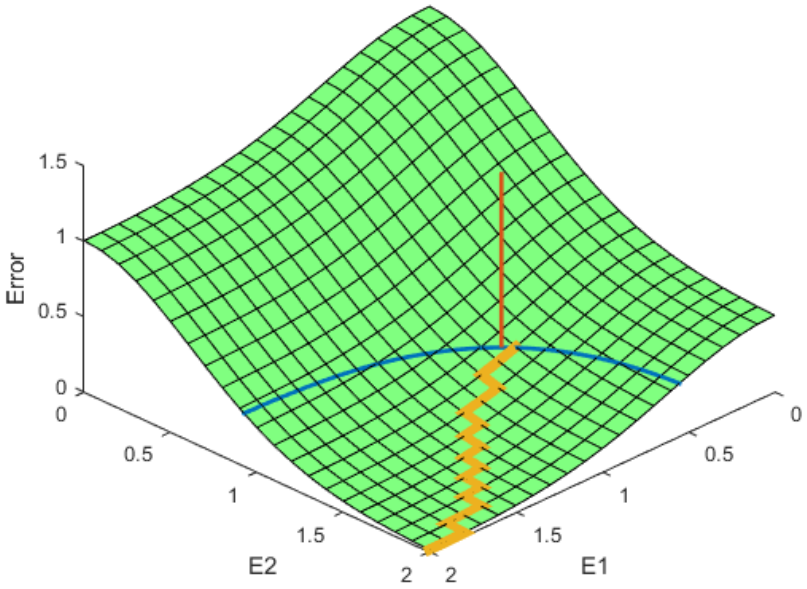


Figure 4.12: The path from maximum energy consumption to the optimal energy distribution for an error of 0.30.

4.3 Conclusion on using PCMOS in a design

Using PCMOS in a complex design requires both accurate estimations of the expected error for a system and an optimal algorithm for finding the optimal energy distribution for that error. Existing models of PCMOS systems ignore the effects of errors due to missed deadlines and propagation delay. This causes an unrealistic estimation of the performance of a PCMOS system, even when the E-P curves of a building block is known for the situation that they are isolated. Existing models do offer precise descriptions of the propagation of errors through a PCMOS system.

The algorithm to assign different energy levels to different parts of a digital circuit that was given in this chapter provides an easy way to approximate the optimal energy distribution for a system, but not for the entire problem space. Its optimality cannot be proven for every situation and it has even been proven that a situation exist where it cannot find the optimal distribution at all. In order to create an algorithm that can find the optimal solution for any situation, a better understanding of the problem of finding an optimal energy distribution is required. The analysis in this chapter should be expanded in further research in order to find an optimal algorithm.

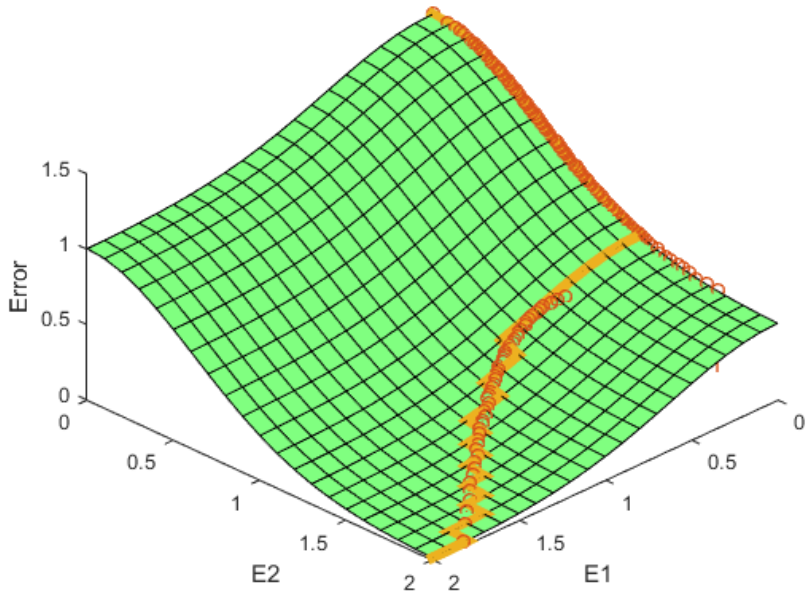


Figure 4.13: Optimal energy distribution per height (red dots) and path chosen by algorithm.

5 Conclusion

Existing models of probabilistic CMOS (PCMOS) are inaccurate in their prediction of Energy-Probability (E-P) curves. They only provide an accurate prediction in situations where the output signal is allowed enough time to settle before the sampling moment. In the sub-threshold region of CMOS transistors, the operating frequency should be lowered exponentially with the supply voltage in order to meet this requirement. Lowering the operating frequency of a PCMOS inverter effectively shifts the probabilistic area of the E-P curve to lower V_{dd} (the transition from 0.5 to 1 is shifted). The same effect can also be achieved by increasing the width of a transistor, which also decreases noise. Less noise results in a shorter probabilistic region (the transition from 0.5 to 1 probability becomes steeper). Higher temperature also increases noise, but at the same time also lowers the threshold voltage.

Using PCMOS in a complex design requires both accurate estimations of the expected error for a system and an optimal algorithm for finding the optimal energy distribution for that error. Existing models of PCMOS systems ignore the effects of errors due to increasing propagation delay at low supply voltage. This causes an unrealistic estimation of the performance of a PCMOS system, even when the E-P curves of a building block are known for the situation that they are isolated. No algorithms exist that guarantee finding the optimal distribution of energy for a specific error. In order to create an algorithm that can find the optimal solution for any situation, a better understanding of the problem of finding an optimal energy distribution is required.

6 Recommendations

This report ended with the description of an algorithm that can approximate the optimal energy distribution for a probabilistic system, though not for every situation. Such an algorithm massively reduces the amount of work it takes to design an optimal probabilistic system. However, much more information about the interaction between probabilistic building blocks is needed before such an algorithm can be attempted. This should be the main focus of future research on this subject.

The analysis of the PCMOS inverter covered some important parameters, but many more parameters exist that influence the performance of probabilistic CMOS. One of the influences that was only briefly mentioned is the difference in supply voltage levels between elements. Due to different interpretations of the same voltages it can be expected that the performance of a system will decrease. It should be researched how this difference influences performance.

Another factor that was not analyzed is the balancing between N-MOS and P-MOS. The assumption during this report has always been that the performance for a rising edge is the same as that for a falling edge. If the N-MOS and P-MOS are not balanced, this is not the case. The average performance may drop due to such imperfections.

The simulations that were presented in this report can be improved upon. Making the input sources random instead of a fixed pattern will create more realistic E-P curves. This reduces the errors that may have been introduced into the data in this research due to repetitive input patterns. Any future simulations should use randomized input sources to prevent these errors and make the simulations even more realistic.

In order to make these kinds of systems feasible for production in the future, a limited number of supply voltage levels should be used. This technique is called Voltage Binning. In most existing research an unlimited amount of unique supply voltage levels are allowed. Adding a restriction on this adds a new dimension to the problem of finding the optimal distribution, making it even more complex, but also improves the feasibility of implementing a PCMOS system.

Bibliography

[1] L. B. Kish, "End of moore's law: thermal (noise) death of integration in micro and nano electronics," *Physics Letters A*, vol. 305, no. 3, pp. 144–149, 2002.

[2] ITRS, "Overall Roadmap Technology Characteristics (ORTC) Table," tech. rep., International Technology Roadmap for Semiconductors, 2013.

[3] L. Kish, "Moore's law and the energy requirement of computing versus performance," in *Circuits, Devices and Systems, IEE Proceedings-*, vol. 151-2, pp. 190–194, IET, 2004.

[4] G. Roll, J. Mo, E. Lind, S. Johansson, and L.-E. Wernersson, "Defect evaluation in ingaas field effect transistors with hfo2 or al2o3 dielectric," *Applied Physics Letters*, vol. 106, no. 20, p. 203503, 2015.

[5] C. T. Chow, L. S. M. Tsui, P. H. W. Leong, W. Luk, and S. J. Wilton, "Dynamic voltage scaling for commercial fpgas," in *Field-Programmable Technology, 2005. Proceedings. 2005 IEEE International Conference on*, pp. 173–180, IEEE, 2005.

[6] K. Palem, "Energy aware computing through randomized switching," tech. rep., Technical Report GIT-CC-03-16, Georgia Institute of Technology, 2003.

[7] Various, "The error function." `https://en.wikipedia.org/wiki/Error_function`. Accessed: 2015-7-12.

[8] K. V. Palem, "Energy aware computing through probabilistic switching: A study of limits," *Computers, IEEE Transactions on*, vol. 54, no. 9, pp. 1123–1137, 2005.

[9] N. S. Kim, T. Austin, D. Baauw, T. Mudge, K. Flautner, J. S. Hu, M. J. Irwin, M. Kandemir, and V. Narayanan, "Leakage current: Moore's law meets static power," *computer*, vol. 36, no. 12, pp. 68–75, 2003.

[10] S. Cheemalavagu, P. Korkmaz, K. V. Palem, B. E. Akgul, and L. N. Chakrapani, "A probabilistic cmos switch and its realization by exploiting noise," in *IFIP International Conference on VLSI*, pp. 535–541, 2005.

[11] M. S. Lau, K. V. Ling, Y. C. Chu, and A. Bhanu, "Modeling of probabilistic ripple-carry adders," in *Electronic Design, Test and Application, 2010. DELTA'10. Fifth IEEE International Symposium on*, pp. 201–206, IEEE, 2010.

[12] M. S. Lau, K.-V. Ling, and Y.-C. Chu, "Energy-aware probabilistic multiplier: design and analysis," in *Proceedings of the 2009 international conference on Compilers, architecture, and synthesis for embedded systems*, pp. 281–290, ACM, 2009.

[13] J. Kim and S. Tiwari, "Inexact computing using probabilistic circuits: Ultra low-power digital processing," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 10, no. 2, p. 16, 2014.

[14] A. H. M. v. R. P. R. van der Meer, A. van Staveren, *Low-Power Deep Sub-Micron CMOS Logic: Subthreshold Current Reduction*. Springer, Dordrecht, 2004.

[15] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in *Test Symposium (ETS), 2013 18th IEEE European*, pp. 1–6, IEEE, 2013.

[16] Z. Prijić, S. Dimitrijev, and N. Stojadinović, "Analysis of temperature dependence of {CMOS} transistors' threshold voltage," *Microelectronics Reliability*, vol. 31, no. 1, pp. 33 – 37, 1991.

[17] J. George, B. Marr, B. E. Akgul, and K. V. Palem, "Probabilistic arithmetic and energy efficient embedded signal processing," in *Proceedings of the 2006 international conference on Compilers, architecture and synthesis for embedded systems*, pp. 158–168, ACM, 2006.

[18] P. Bromiley, "Products and convolutions of gaussian probability density functions." `http://www.tina-vision.net/docs/memos/2003-003.pdf`. Accessed: 2016-03-31.