

# Ontwerp van een RRAM geheugen

Wouter Diels  
Alexander Standaert

Thesis voorge dragen tot het behalen  
van de graad van Master of Science  
in de ingenieurswetenschappen:  
elektrotechniek, optie Elektronica en  
geïntegreerde schakelingen

**Promotor:**

Prof. dr. ir. W. Dehaene

**Assessoren:**

Prof. dr. ir. R. Lauwereins  
Prof. dr. ir. M. Verhelst

**Begeleiders:**

ir. B. Baran  
dr. ir. S. Cosemans

© Copyright KU Leuven

Zonder voorafgaande schriftelijke toestemming van zowel de promotor als de auteurs is overnemen, kopiëren, gebruiken of realiseren van deze uitgave of gedeelten ervan verboden. Voor aanvragen tot of informatie i.v.m. het overnemen en/of gebruik en/of realisatie van gedeelten uit deze publicatie, wend u tot ESAT, Kasteelpark Arenberg 10 postbus 2440, B-3001 Heverlee, +32-16-321130 of via e-mail [info@esat.kuleuven.be](mailto:info@esat.kuleuven.be).

Voorafgaande schriftelijke toestemming van de promotor is eveneens vereist voor het aanwenden van de in deze masterproef beschreven (originele) methoden, producten, schakelingen en programma's voor industrieel of commercieel nut en voor de inzending van deze publicatie ter deelname aan wetenschappelijke prijzen of wedstrijden.

# Voorwoord

Dit is mijn dankwoord om iedereen te danken die mij bezig gehouden heeft. Hierbij dank ik mijn promotor, mijn begeleider en de voltallige jury. Ook mijn familie heeft mij erg gesteund natuurlijk.

*Wouter Diels  
Alexander Standaert*

# Inhoudsopgave

<b>Voorwoord</b>	i
<b>Samenvatting</b>	iv
<b>Lijst van figuren en tabellen</b>	v
<b>Lijst van afkortingen en symbolen</b>	vii
<b>1 Inleiding</b>	1
1.1 Doel en afbakening van dit werk . . . . .	1
1.2 Structuur van de tekst . . . . .	2
<b>2 Geheugencel</b>	3
2.1 Memristor . . . . .	3
2.2 Memristor in een geheugenstructuur . . . . .	6
2.3 Besluit . . . . .	6
<b>3 Geheugenarchitectuur</b>	7
3.1 Van transistorniveau tot systeemniveau . . . . .	7
3.2 Besluit . . . . .	9
<b>4 Lastimpedantie-analyse</b>	11
4.1 algemene last eigenschappen en specificaties . . . . .	11
4.2 evalueren van de last . . . . .	12
4.3 vergelijking van verschillende types last . . . . .	14
<b>5 Sense Amplifier analyse</b>	23
5.1 Types SA . . . . .	23
5.2 Offsetspanning . . . . .	23
5.3 Sensitiviteitsanalyse . . . . .	24
5.4 Paretosimulatie . . . . .	26
5.5 Besluit . . . . .	26
<b>6 Omringende logica</b>	29
6.1 Decoders . . . . .	29
6.2 Buffers . . . . .	32
6.3 BL- en WL-drivers . . . . .	34
6.4 Passgates . . . . .	34
6.5 Besluit . . . . .	34
<b>7 Timing en optimalisatie</b>	35

## INHOUDSOPGAVE

---

7.1	Timing . . . . .	35
7.2	Analyse verschillende geheugenconfiguraties . . . . .	41
<b>8</b>	<b>Besluit</b>	<b>45</b>
<b>A</b>	<b>Charge injectie met ideale spice bronnen.</b>	<b>49</b>
	<b>Bibliografie</b>	<b>51</b>

# Samenvatting

In dit **abstract** environment wordt een al dan niet uitgebreide samenvatting van het werk gegeven. De bedoeling is wel dat dit tot 1 bladzijde beperkt blijft.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

# Lijst van figuren en tabellen

## Lijst van figuren

2.1 Abrupte overgang van hoge weerstand naar lage weerstand voor NiO[1]	4
2.2 Metal-Insulator-Metal structuur[12]	4
2.3 Model van het Pt-TiO <sub>2</sub> -Pt staal [10]	5
2.4 Illustratie van forming,resetting en setting [12]	5
3.1 Een geheugencel en een branch	8
3.2 Een Local Block	9
3.3 Een Global Block	9
4.1 Timing globalblock	12
4.2 Test bench voor de last	13
4.3 De verschillende types last	14
4.4 Bitlijn voltage distributie voor een biasload	16
4.5 Lineaire sweep van switchload	17
4.6 Lineaire sweep van biasload	17
4.7 Lineaire sweep van diodeload	18
4.8 Lineaire sweep van bulkload	18
4.9 Lineaire sweep van switchload	19
4.10 Verschillende oplossingen voor de switchload met variabele lengtes en breettes	20
4.11 Bitlijn voltage distributie voor de finale load	21
5.1 een sense amplifier	23
5.2 Illustratie van offsetspanning	24
5.3 Door $\beta$ -mismatch is ladingsinjectie van de pass-gates niet meer gematched en gaat de SA foutief latchen	25
5.4 Simulatieopstelling voor het RC-latch-effect	26
5.5 Simulatieresultaten voor het RC-latch-effect:de 2 ingangs-uitgangsknopen zijn voorgeladen op 400mV en 380mV. Na 1,6ns wordt de SA aangezet. De SA is ideaal voor deze simulatie.	26
5.6 Vergelijking situatie met voorgeladen (eindige) capaciteit en situatie met spanningsbron (oneindige capaciteit)	27
6.1 Opbouw voor grotere decoders	29

## LIJST VAN FIGUREN EN TABELLEN

---

6.2	basis decoders . . . . .	30
6.3	vergelijking van decoder types . . . . .	32
6.4	Timing globalblock . . . . .	33
6.5	Timing globalblock . . . . .	33
6.6	Timing globalblock . . . . .	34
7.1	Timing problemen bij de bitlijn . . . . .	36
7.2	Globalblock logica . . . . .	37
7.3	Timing globalblock . . . . .	37
7.4	Controle logica memory array . . . . .	38
7.5	Timing controle logica memory array . . . . .	38
7.6	Delay van woordlijn decoders + buffers ifv bitlijn decoders . . . . .	39
7.7	Controle logica memory array . . . . .	39
7.8	Timing controle logica memory array . . . . .	40
7.9	logica rond SA . . . . .	40
7.10	Timing logica rond SA . . . . .	41
7.11	Timing controle logica memory array . . . . .	42
7.12	Timing controle logica memory array . . . . .	43
7.13	Timing controle logica memory array . . . . .	44
A.1	Timing globalblock . . . . .	50
A.2	Timing globalblock . . . . .	50

## Lijst van tabellen

6.1	dfs . . . . .	31
6.2	qsdqsdq . . . . .	34

# Lijst van afkortingen en symbolen

## Afkortingen

BL	Bit Line
CDF	Cumulative Distribution Function
GB	Global Block
LB	Local Block
NoBLpLB	Number of Bit Lines per Local Block
NoGB	Number of Global Blocks
NoWLpB	Number of Word Lines per Branch
PDF	Probability Distribution Function
RAM	Random Access Memory
RRAM	Resistive Random Access Memory
SA	Sense Amplifier
SL	Source Line
WL	Word Line
MTJ	Magnetic tunnel junction
HRS	High resistant state
LRS	Low resistant state

## Symbolen

- 42 “The Answer to the Ultimate Question of Life, the Universe, and Everything”



# **Hoofdstuk 1**

## **Inleiding**

Vandaag de dag is elektronica niet meer uit het leven weg te denken. Van de smartphone tot het digitaal horloge, van de bordcomputer in de moderne wagen tot de microprocessor in de vaatwasser, overal vind je wel elektronica terug. Sinds Gordon Moore ongeveer 50 jaar geleden de uitspraak deed dat het aantal transistoren op eenzelfde oppervlakte per twee jaar zou verdubbelen [7], is de industrie er over het algemeen goed in geslaagd dit te verwezenlijken. Dit leidde tot de snelle en uiterst complexe chips die we vandaag allemaal goedkoop aankopen.

Naarmate de processorkracht groter werd, steeg ook de vraag voor grotere en snellere geheugens om deze processorkracht ook effectief uit te buiten. Static Random Access Memory (SRAM) blijft een populaire keuze voor snelle ingebedde geheugens, maar heeft het nadeel vluchtig te zijn: eenmaal de voedingsspanning wordt afgeschakeld, verdwijnt de informatie. Flash-geheugens, door veel mensen gebruikt voor massa-opslag in USB-sticks of SSDs, hebben ook hun weg gevonden tot het ingebedde domein en behoren wel tot de klasse van niet-vluchige geheugens. Het blijkt echter bijzonder moeilijk om flash-geheugens verder te verkleinen [8].

Onderzoek naar nieuwe geheugens is dan ook onontbeerlijk. Zo zijn er al nieuwe nieuwe kandidaten in opmars die hoopgevende tekens geven om te concurreren met (ingebedde) flash-geheugens. MRAMs (Magnetic RAMs) en in het bijzonder STT-RAM (Spin-Transfer Torque) zullen op termijn een belangrijke rol gaan spelen.

Een andere kandidaat is Resistive RAM (RRAM of ReRAM). Daar waar SRAM-en flash-cellen de informatie bevatten via het al dan niet aanwezig zijn van lading, bevat een RRAM-cel informatie door een bepaalde elektrische weerstand aan te nemen. RRAM zou geen problemen hebben om nog even op de klassieke manier mee te schalen en is dus zonder meer een interessante piste om te onderzoeken. Bovendien zou het gefabriceerd kunnen worden met goedkopere processen dan flash-geheugens - bij flash-geheugenfabricatie zijn vaak dure extra maskers vereist.

### **1.1 Doel en afbakening van dit werk**

Dit werk beschrijft het ontwerp van een 4MByte RRAM-geheugen voor ingebedde toepassingen. De doelstelling is een pareto-optimaal (dynamische energie-snelheid-

## 1. INLEIDING

---

oppervlakte) werkend circuit te ontwerpen, gewapend tegen variabiliteit - ongecorreleerde gedragsvariaties van componenten. Het ontwerp houdt rekening met data-retentie bij het uitlezen van bits. De analyse focust op de leesbewerking, de schrijfbewerking valt buiten het bereik van dit werk. [Er worden wel mogelijke oplossingen aangereikt, maar deze werden niet uitdrukkelijk onderzocht]

Voor de leesbewerking wordt het geheugen-element gemodelleerd als een weerstand waarvan de nominale weerstandswaarde afhangt van de celtoestand. Wanneer variabiliteit wordt onderzocht, zal deze weerstandswaarde een stochastische variabele worden met een normale verdeling.

Temperatuursvariaties werden niet in rekening genomen, maar aangezien dit een globale variabele is en het systeem differentieel werkt, wordt niet verwacht dat de performantie aanzienlijk zal verminderen.

Alle data die worden getoond, komen voort uit Spectre-simulaties met 45nm PTM transistormodellen.

## 1.2 Structuur van de tekst

In hoofdstuk 2 zal de technologie van een RRAM geheugen uiteengezet worden, alsook diens toepassingen. Ook zal het elementaire principe om uit een weerstand een nuttig elektrisch signaal te vormen uitgelegd worden. In hoofdstuk 3 wordt het geheugensysteem vanuit vogelperspectief besproken. Er wordt hier ook aangehaald wat de tunebare parameters zijn van de architectuur. Voor een robuuste, snelle en laag-energetische leesoperatie uit te voeren is het belangrijk het geheugenelement te combineren met een zorgvuldig gekozen impedantie, dit wordt onderzocht in hoofdstuk 4. Uiteindelijk zal er een bitstream moeten gegenereerd worden aan de uitgang van het systeem, de sense amplifier zorgt hiervoor en wordt besproken in hoofdstuk 5. In de geheugenstructuur zijn ook bepaalde logische (digitale) operaties nodig om op basis van het opgegeven adres de juiste cel aan te spreken, de hiervoor gebruikte blokken worden beschreven en geanalyseerd in hoofdstuk 6. Ten slotte zal in hoofdstuk 7 de timing van controlesignalen onderzocht worden en hoe het systeem te optimalizeren door middel van de architectuurparameters te tunen.

# Hoofdstuk 2

## Geheugencel

Elk geheugen bestaat uit een verzameling individuele cellen die op een of andere manier informatie bevatten. In dit hoofdstuk wordt eerst dieper ingegaan op de manier waarom een R-RAM geheugencel informatie bevat en vervolgens hoe deze informatie [elektrisch] kan worden uitgelezen.

### 2.1 Memristor

Het essentiële element van een R-RAM geheugencel is ontegensprekbaar de zogenaamde memristor. De memristor wordt ook wel gezien als de 4<sup>e</sup> passieve component, naast de weerstand, spoel en condensator.

#### 2.1.1 Theoretisch principe

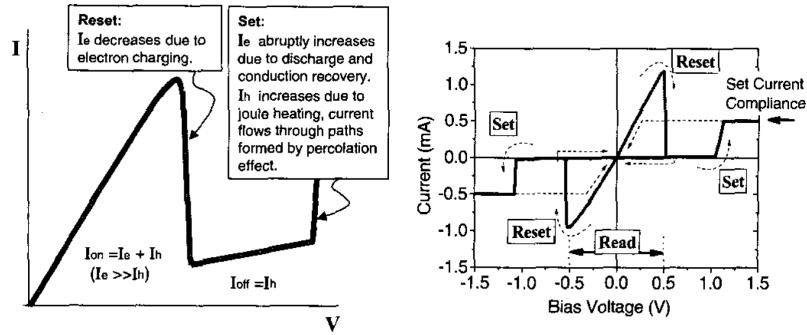
In 1971 publiceerde Leon Chua, een onderzoeker in o.a. niet-lineaire circuittheorie, een artikel waarin hij opmerkte dat er voor de 4 fundamentele circuitvariabelen (de spanning  $v$ , stroom  $i$ , lading  $q$  en flux  $\phi^1$ ) van 6 mogelijke onderlinge relaties er slechts 5 gekend waren:  $q(t) = \int_{-\infty}^t i(\tau) d\tau$ ,  $\phi(t) = \int_{-\infty}^t v(\tau) d\tau$ ,  $v(t) = R * i(t)$ ,  $q(t) = C * v(t)$  en  $\phi(t) = L * i(t)$  volgen uit de wetten van Maxwell en uit de definities van de weerstand, spoel en condensator, maar er ontbrak een relatie tussen  $\phi$  en  $q$ .<sup>[3]</sup> Hij suggereerde dat er een 4e nog niet ontdekte passieve 2-pool moest bestaan die dit verband herbergde. Hij stelde dat  $M(q) = \frac{d\phi(q)}{dq}$  met  $M$  de *memristance*. Hieruit volgt dat voor dit element  $v(t) = M(q(t))i(t)$ . Indien er een lineair verband bestaat tussen  $\phi$  en  $q$ , gedraagt dit element zich als een gewone weerstand. Enkel wanneer er een niet-lineair verband bestaat, beginnen er zich interessante karakteristieken voor te doen. Zo gedraagt het element zich ogenblikkelijk als een weerstand, maar gaat deze weerstandswaarde variëren in de tijd aan de hand van de stroom die er doorgelopen heeft. Gebaseerd op deze conclusie doopte hij deze component de memristor (een contractie van memory en resistor). Chua beëindigde zijn artikel met te erkennen dat er op dat moment nog geen fysische memristor was ontdekt, maar dat dit in de toekomst wel kon gebeuren, al dan niet zelfs per ongeluk. Hij gaf zelfs aan dat er

---

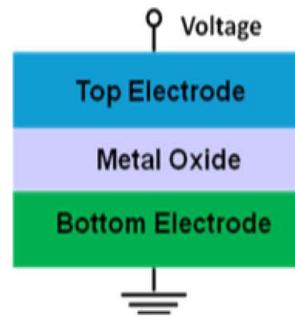
<sup>1</sup> $\phi(t) = \int_{-\infty}^t v(\tau) d\tau$ , voor een ideale inductantie is dit hetzelfde als magnetische flux

## 2. GEHEUGENCSEL

---



Figuur 2.1: Abrupte overgang van hoge weerstand naar lage weerstand voor NiO[1]



Figuur 2.2: Metal-Insulator-Metal structuur[12]

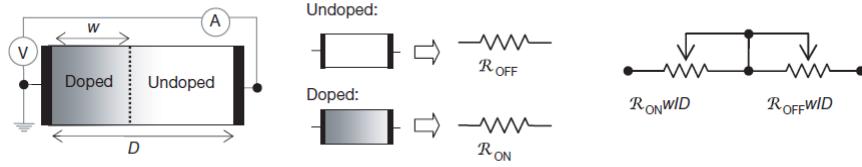
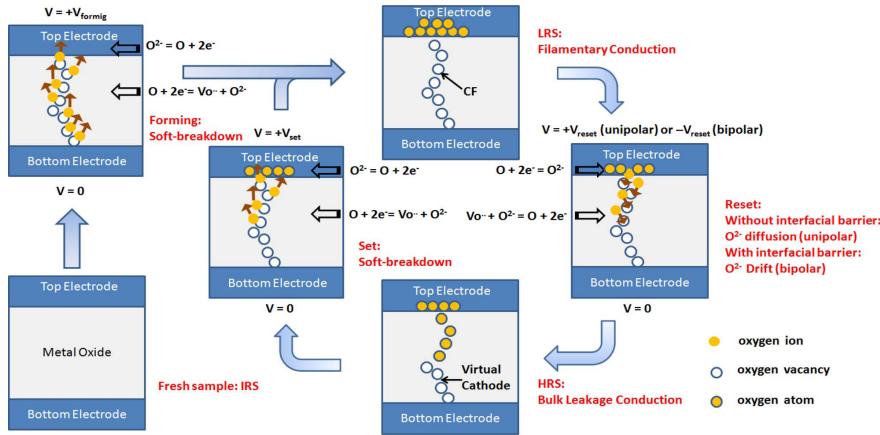
misschien al in die tijd materialen met memristorkarakteristieken gebruikt werden, maar dat men hier over keek. Hij zou gelijk krijgen.

### 2.1.2 Fysische memristors

Er werd reeds langer (zelfs al sinds de jaren 60) opgemerkt dat sommige metaaloxides, die normaal gezien als elektrisch isolator functioneren, een plotselinge overgang kunnen ervaren naar een veel geleidender staat (zie figuur 2.1). Dit gebeurt veelal in een configuratie waarbij het oxide wordt geplaatst tussen 2 metalen (MIM configuratie).[12] (zie ook figuur 2.2)

In 2008 publiceerde een onderzoeksgrond van Hewlett-Packard een artikel waarin ze opmerkten dat het gedrag van hun Pt-TiO<sub>2</sub>-Pt stalen een merkwaardige gelijkenis vertoonde met Chua's memristor.[10] Uit de modellering van hun stalen argumenteerden ze dat dit een ideale memristor zou zijn en dat het effect meer uitgesproken is bij kleine afmetingen: het titaniumoxide bestaat uit 2 delen: zuiver TiO<sub>2</sub>, een halfgeleider met hoge weerstand, en TiO<sub>2-x</sub> met zuurstofafwezigheid (oxygen vacancies) met een veel lagere weerstand. Door een elektrisch veld aan te leggen worden zuurstofatomen weg of in het rooster getrokken en verandert de verhouding TiO<sub>2</sub> en TiO<sub>2-x</sub> en dus ook de netto weerstand (zie figuur 2.3).

Voor deze opstelling geldt dan dat  $M(q) = R_{off}(1 - \frac{\mu_v R_{on}}{D^2} q(t))$  met D de dikte

Figuur 2.3: Model van het Pt-TiO<sub>2</sub>-Pt staal [10]

Figuur 2.4: Illustratie van forming,resetting en setting [12]

van de titaniumoxidefilm,  $\mu_v$  de mobiliteit van de zuurstofionen en  $R_{on} \leq M \leq R_{off}$ . Het dynamisch gedrag van de ogenblikkelijke weerstandswaarde is dus afhankelijk van het verloop van de stroom in de tijd en dit des te meer in het nanometerdomein.

Naast titaniumoxide zijn er nog andere materialen die schakelend weerstandsverdrag vertonen zoals nikkeloxide[1], hafniumoxide[2], aluminiumoxide[5],... Niet altijd kunnen de resultaten gemodelleerd worden volgens de originele memristortheorie, maar desalnietemin zullen deze materiaalconfiguraties bruikbaar zijn in toepassingen. Bij al deze MIM-configuraties blijft het mechanisme wel hetzelfde: na fabricatie is het oxidekristal intrinsiek zuiver, maar onder druk van een voldoende groot elektrisch veld zullen de zuurstofatomen losgerukt worden uit het rooster naar de anode. Het gebrek aan zuurstofatomen zorgt voor conductieve filamenten. Het element bereikt dan een laagresistieve staat (LRS). Deze zachte doorslag van het zuivere oxide wordt *forming* genoemd. Het proces is tot zekere hoogte omkeerbaar (*reset*), maar er zullen altijd meer defecten in het kristal zijn dan na fabricatie. Dit betekent dus ook dat nadat de memristor één keer een forming- en resetproces is ondergaan en zich terug in een hoogresistieve staat (HRS) bevindt, er hierna een minder groot elektrisch veld nodig is om terug tot een LRS te komen. Dit proces heet *setting*. Deze drie processen zijn geïllustreerd op figuur 2.4.

De verschillende MIM-structuren hebben ook verschillende eigenschappen. Zo moet er onderscheid gemaakt worden tussen unipolaire schakelen en bipolaire. Bij

## **2. GEHEUGENCEL**

---

bipolair resistief schakelen zal forming/setting optreden wanneer de aangelegde spanning een bepaalde polariteit heeft en resetting bij de omgekeerde polariteit. Bij unipolair schakelen is de amplitude van de spanning doorslaggevend voor welke van de 3 processen zal optreden, niet de polariteit.

### **2.1.3 Toepassingen**

Naast uiteraard de geheugentoepassing, die in de volgende sectie ingeleid zal worden, hebben resistief schakelende elementen nog andere toepassingen. Met de resistief chakelende elementen zou een *crossbar latch* gerealiseerd kunnen worden,

## **2.2 Memristor in een geheugenstructuur**

In dit werk wordt gebruik gemaakt van een *1 Transistor, 1 Resistor* (maar eigenlijk dus een memristor) architectuur, de combinatie van deze twee vormt de geheugencel, maar er zijn nog een paar andere configuraties die zouden toegepast kunnen worden.

### **2.2.1 1T1R**

### **2.2.2 1R**

### **2.2.3 1T1D**

## **2.3 Besluit**

# **Hoofdstuk 3**

## **Geheugenarchitectuur**

De afzonderlijke geheugencellen zullen uiteindelijk samengebracht moeten worden in een geheel. In dit hoofdstuk zal de algemene structuur besproken worden alsook de vrijheidsgraden die in hoofdstuk 7 onderzocht worden voor een optimaal werkend systeem. Ten slotte zullen ook nog de bouwblokken aangekaard worden die meer uitvoerig besproken worden in de volgende hoofdstukken.

### **3.1 Van transistorniveau tot systeemniveau**

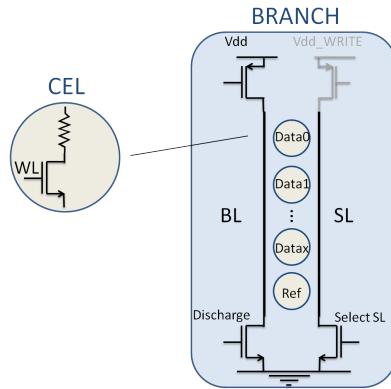
Hoewel het hele systeem op de chip in weze bestaat uit transistoren en passieve componenten, zal dit nog onmogelijk te begrijpen vallen op zulke grote schaal. Het is daarom aangewezen om meer abstractie te maken en de componenten vanop een hoger niveau te bekijken.

#### **3.1.1 Cel**

Zoals besproken in hoofdstuk 2 zal dit het bouwblok zijn dat het vaakst terug zal te vinden zijn op het geheugensysteem. De cel bestaat uit een memristor en een transistor. De geheugencel heeft drie terminals: de gate van de transistor, die zal verbonden worden met een wordline, de source van de transistor, die zal verbonden worden met een sourceline en tenslotte de terminal van de memristor, die zal verbonden worden met een bitline. Een cel wiens memristor zich in een willekeurige resistieve staat bevindt is een datacel, terwijl de memristor van referentiecellen in een voorgeprogrammeerde en dus gekende resistieve staat verkeert.

#### **3.1.2 Branch**

In een branch worden er een bepaald aantal datacellen verbonden aan één BL en één SL. Dit aantal wordt *Number of Word Lines per Branch* (NoWLpB) genoemd en is een van de vrijheidsgraden van de geheugenarchitectuur. Naast alle datacellen is er ook nog één referentiecel verbonden aan de BL en SL van de branch. Elke BL wordt via een pMOS-transistor verbonden met de voedingsspanning Vdd en via een nMOS-transistor aan de grondspanning Vss. In dit werk is er enkel een nMOS-transistor



Figuur 3.1: Een geheugencel en een branch

die de SL verbindt met Vss.<sup>1</sup> De nMOS-transistoren aan BL en SL fungeren als schakelaars, de pMOS-transistor wordt gebruikt als impedantie voor een resistieve spanningsdeling (zie hoofdstuk 4). Ter illustratie wordt de samenhang tussen cel en branch getoond in figuur 3.1.

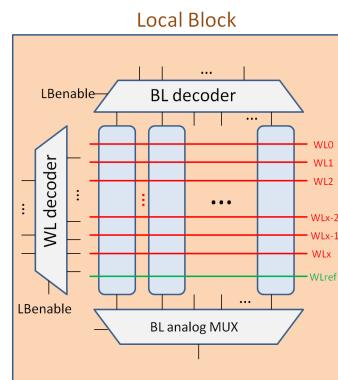
### 3.1.3 Local Block

Verschillende BLs en SLs worden samengebracht in een Local Block, waarvan de vrijheidsgraad *Number of Bit Lines per Local Block* (NoBLpLB) heet. In een LB bevinden er zich dus NoBLpLB x NoWLpB datacellen en NoBLpLB referentiecellen. Ook zitten er in een Local Block zowel BL- als WL-decoders. De structuur van een Local Block is geïllustreerd op figuur 3.2. De uitgangen van de WL-decoder zullen (mits wat buffering tussenin) worden aangesloten aan de data-WLs zelf, die van de BL-decoder zullen een spanningsdeling teweeg brengen op de BLs. De referentie-WL zal via een extern signaal verbonden worden. Aangezien een LB zowel data- als referentiecellen bevat, gaat een LB twee werkingsmodes hebben: een mode waarbij er één datacel wordt aangesproken en een mode waarbij er een bepaald aantal referentiecellen in parallel wordt aangesproken.

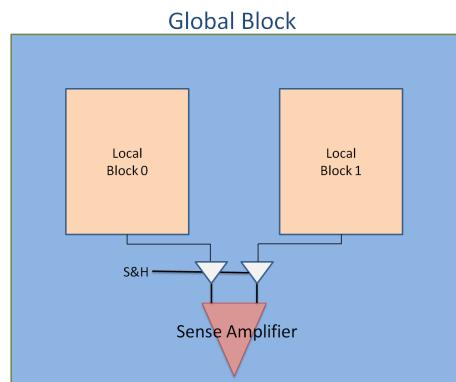
### 3.1.4 Global Block

Een Global Block bestaat uit twee LBs en een sense amplifier (SA). In het ene LB gaat er een datasignaal geproduceerd worden, in het andere een referentiesignaal (zie figuur 3.3). Vervolgens gaat de SA dit kleine signaalverschil versterken tot een zuivere rail-to-rail output. Aan de uitgang van het GB verschijnen dan ook de opgevraagde bits. De laatste architectuurvrijheidsgraad is de *Number of Global Blocks* (NoGB), het totale geheugen bevat dus NoGB x 2 x NoBLpLB x NoWLpB geheugencellen.

<sup>1</sup>In een volledig geheugensysteem zou de SL via een pMOS ook nog verbonden zijn met een niet onderzochte spanningsknoop Vdd\_write. De pMOS zou dan worden aangezet voor schrijfwerking.



Figuur 3.2: Een Local Block



Figuur 3.3: Een Global Block

## 3.2 Besluit



## Hoofdstuk 4

# Lastimpedantie-analyse

Om een cel uit te lezen wordt er een spanning gegenereerd op de bitline door middel van spanningsdeling. Het is dus belangrijk om de 2 impedanties van de spanningsdeler zodanig te kiezen voor optimale snelheid, bitline spanningsverschil en memristorretentie. Ook belangrijk is dat deze impedanties robuust zijn tegen variabiliteit.

### 4.1 algemene last eigenschappen en specificaties

In deze eerste sectie bestuderen we de combinatie van last en memristor cell als een heel simpel model namelijk twee weerstanden in serie (zie figuur ??). Dit om aan te tonen dat de waarde weerstands waarde van de last een grote invloed heeft op de het voltage verschil tussen een hoge en lage cel weerstand, bitlijn snelheid en de sensitivity van beide. Het verschil in bitlijn voltage tussen een hoge en lage cell is van belang voor de toleraties op de referentie voltage en sense amplifier mismatch. In ons simple model kan het verschil in bitlijn voltage analitisch berekend worden met de volgende formule:

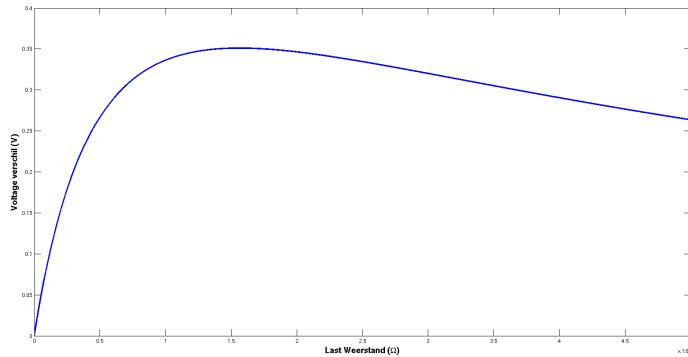
$$\Delta V = \frac{R_{HRS}}{R_{last} + R_{HRS}} - \frac{R_{LRS}}{R_{last} + R_{LRS}} \quad (4.1)$$

Voor constante waarden van  $R_{HRS}$  en  $R_{LRS}$  zal er een maximum zijn in  $\Delta V$  zoals duidelijk gezien kan worden in figuur 4.1. De sensitiviteit van de last weerstand op het spannings verschil moet men voorzichtig interpreteren. Op figuur 4.1 kan gezien worden dat de helling voor de piek stijker is als na de piek. Het is dus beter om een iets grotere weerstand te hebben als een iets te kleine weerstand. Maar als men de weerstand naar transistor afmetingen vertaalt, kan met dit op verschillend manieren realiseren. Een grote weerstand realiseren met een transistor met minimale lengte, zal betekenen dat de breite van de transistor klein moet zijn. Dit zal dan gevoeliger zijn voor mismatch dan een grotere breite van transistor.

De snelheid van het opladen van de bitlijn kan in het simple model ook analitisch geschreven worden. De volgende vergelijking stelt de tijd voor waar de bitlijn 99% is opgeladen

$$t = -\ln(0.01) * RC \quad (4.2)$$

$$R^{-1} = \frac{1}{R_{cell}} + \frac{1}{R_{last}} \quad (4.3)$$



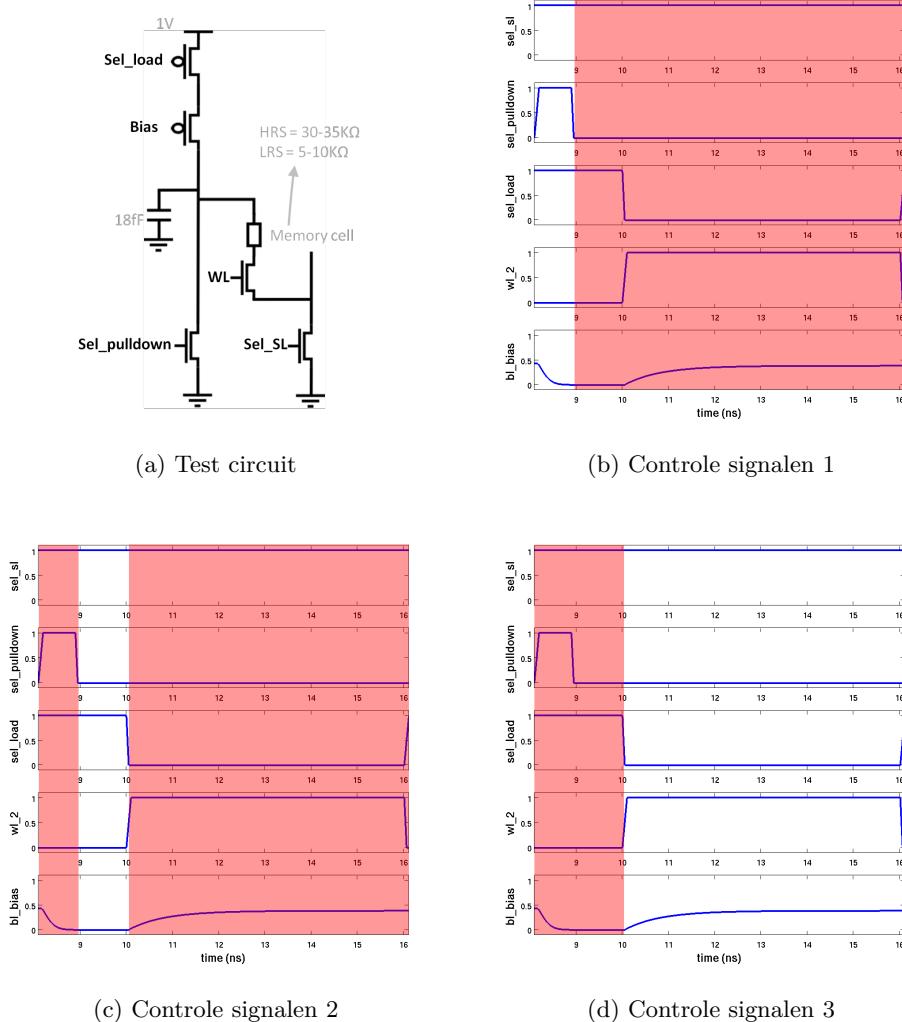
Figuur 4.1: Timing globalblock

## 4.2 evalueren van de last

Om verschillende lasten met elkaar te kunnen vergelijken, is het belangrijk om hun eigenschappen allemaal op dezelfde manier te bekomen. Figuur 4.2 geeft de verschillende aspecten van de gebruikte simulatie setup weer. Het test circuit (Figuur 4.2a) stelt een bitlijn voor met een capaciteit van 18fF, wat ruiw weg overeenkomt met een bitlijn met 100 cellen op. Aan deze bitlijn zijn een last, een ontlaad transistor en een memristor weerstand aangesloten. De ontlaad transistor is minimaal gehouden. De memristor weerstand kan de volgende waarden hebben: tussen  $5k\Omega$  en  $10k\Omega$  voor de LRS, tussen  $30k\Omega$  en  $35k\Omega$  voor de HRS. De nominale waarden voor LRS en HRS zijn  $7.5k\Omega$  en  $32.5k\Omega$ . Tijdens monte-carlo analyses worden dan deze nominale waarden als gemiddelde van een gausische distributie genomen met  $\sigma = 0.833k\Omega$ . Aan deze memristor weerstand hangt een select transistor, die ook minimaal gehouden wordt, en de combinatie van deze word de geheugen cell genoemd. Aan deze geheugen cell hangt nog een selectlijn transistor met een breedte van 500nm. Deze transistor werd bewust groot gemaakt om de totale weerstand in de onderste tak voornamelijk te laten afhangen van de geheugen cell. Aan deze selectlijn werd er ook een weerstand van 18fF aan gehangen, deze doet echter niet veel aangezien de select transistor altijd aan gelaten wordt. Tenslotte word de voedings spanning altijd op 1V gehouden.

Figuren 4.2b tot 4.2d stellen de sequentie van alle controle signalen uit tijden de simulatie. Eerst wordt de bitlijn volledig ontladen (4.2b). Vervolgens is er een interval waar niets gebeurt (4.2c) en tenslotte wordt te last aangesloten en de bitlijn

op geladen (4.2d). De simulatie stopt als de bitlijn volledig is opgeladen.



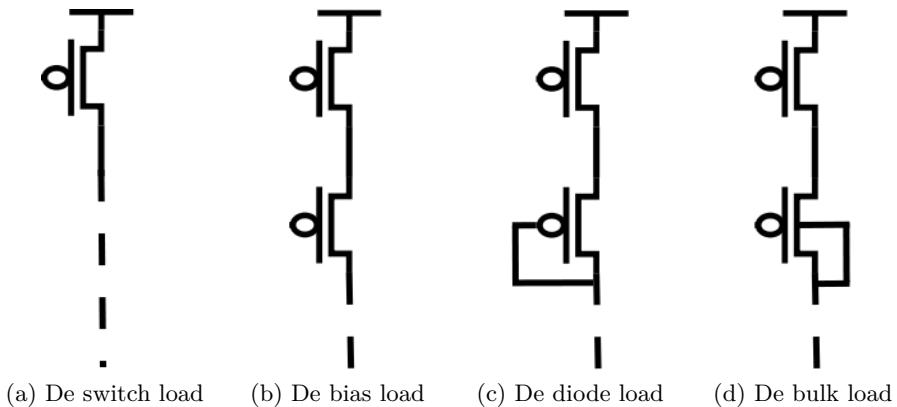
Figuur 4.2: Test bench voor de last

Eens een last gesimuleert is wordt het beoordeelt op het vlak van oppervlakte, bitlijn risetime, nominaal bitlijn voltage verschil en voltage drop over de cell. Het oppervlakte wordt berekend op basis van de lengtes en breedtes van de last transistoren. De bitlijn risetime is de tijd dat nodig is om de bitlijn 99% op te laden. Het nominale bitlijn voltage verschil is het verschil in bitlijn voltage tussen een cell in HRS en LRS, wanneer de bitlijn 100% is opgeladen. De bitlijn wordt veronderstelt 100% opgeladen te zijn op het einde van de simulatie en de simulatie tijd word voldoende lang gehouden om dit te garenderen. De spanningsval over de cell is belangrijk opdat de cell in van state wisselt gedurende de leescyclus. Aangezien de cell voorgesteld wordt

met een weerstand zal dit natuurlijk nooit gebeuren maar dit is wel belangrijk moest er met echte memristors worden gewerkt. De numerieke waarde van de maximale spannings val over de cell is heel erg afhankelijk van het type memristor. In dit onderzoek wordt er gewerkt met een maximum van 0.5V over de cell.

### 4.3 vergelijking van verschillende types last

Voor dit onderzoek worden vier mogelijke kandidaten van last vergeleken: de switchload ([4.3a](#)), de biasload ([4.3b](#)), de diodeload ([4.3c](#)) en de bulkload ([4.3d](#))[\[9\]](#). Eerst wordt er een lineare sweep gedaan op verschillende lasten ([4.3.1](#)), waarbij enkel de breedtes en bias spanningen worden geswept. De lengtes van de transistoren word minimaal gehouden om er voor te zorgen dat de transistoren binnen de pitch van de bitlijn passen. Eens variabiliteit wordt toegevoegd aan de simulatie onder de vorm van monte-carlo ([4.3.2](#)), zal echter blijken dat er het verschil in bitlijn voltage te klein is, en zal de lengte van de last transistoren ook moeten worden vergroot ([4.3.3](#)).



Figuur 4.3: De verschillende types last

#### 4.3.1 Lineaire sweep op de lasten

De switchload bestaat uit één pmos transistor dit volledig wordt aan of afgesloten. Een lineaire sweep met een breedte van de transistor tussen 100nm en 500nm werd gedaan en is geïllustreerd in figuur [4.5](#). Bij het vergroten van de breedte van de transistor zal de weerstand dalen en het verschil tussen de bitlijnen ook. Als we deze last vergelijken met het simpele model uit sectie [4.1](#), zit de weerstand waarde aan de linker kant van de piek uit figuur [4.1](#). Bij het vergroten van de transistor breedte zal de bitlijn spanning stijgen en de spannings val over de cell dus ook. Verder volgt de risetime ook het simple model uit sectie [4.1](#), waarbij de risetime daalt bij kleinere weerstands waardes.

De biasload, is een last met twee pmos transistoren in serie. Bovenste van de twee wordt als een switch gebruikt en dus volledig aan of af gesloten. De onderste van de twee word op een spanning gebiased. Het voordeel van de biasload is dat men een grotere weerstand kan maken en dus de piek kan bereiken uit figuur 4.1. Dit kan men duidelijk zien op de x-assen van figuur 4.6. Ook hier is de breedtes van de transistoren gesweept tussen 100nm en 500nm. De bias spanning is tussen 0V en 0.4V gesweept. Een hogere bias spanning brengt echter geen nuttige bij drage. Door dat te kleinste weerstand dat met deze last te maken is ,binnen deze sweeprange, net iets groter is als deze van de switch load, is de biasload ook iets trager. De oplossingen waarbij dit het geval is, hebben echter een onbruikbaar verschil in bitlijn voltages. De spanningsval over de cell is vergeleken met de switch load heel wat hoger maar voor de meeste oplossingen ligt het nogaltijd onder de limiet van 0.5V.

De diode load bestaat ook uit twee transistoren, de bovenste wordt net als bij de biasload als een switch gebruikt. De onderste is als een diode geconnecteerde transistor gekoppelt. Uit de sweep resultaten (figuur 4.7) blijkt dat deze last heel snel is maar veel te kleinen bitlijn voltage verschillen heeft om bruikbaar te zijn.

De bulkload werd voor gestelt in de paper van Ren et al. [9] als een goede kandidaat omwille van zijn grote uitgangs impedantie. Deze last bestaat uit een switch transistor en een bulk geconnecteerde transistor. Deze bulk geconnecteerde transistor word op 0V gebiaast aangezien deze de beste resultaten gaf. De breedtes vab de transistoren zijn gesweept tussen 100nm en 500nm. De resultaten van deze sweep zijn geillustreerd in figuur 4.8. In de resultaten kan gezien worden dat deze last zich vergelijkbaar gedraagt als de biaslast. Enkel op het vlak van risetime zijn er oplossingen die beter zijn.

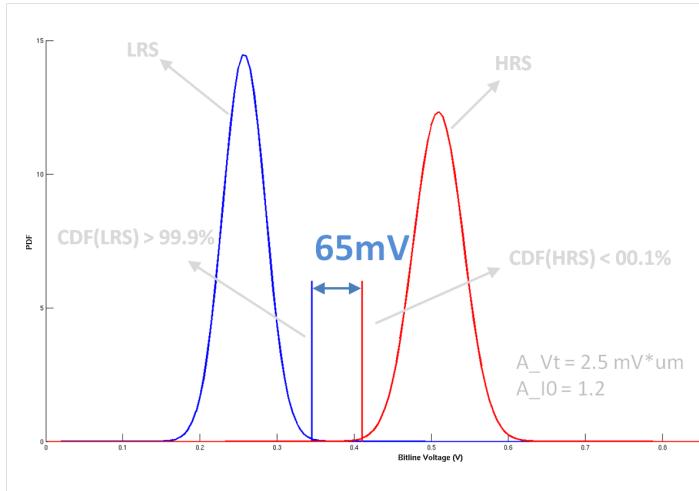
#### 4.3.2 Het toevoegen van variabiliteit

Na een selectie te hebben gemaakt van de oplossingen uit de vorige sectie, worden met deze oplossingen nieuwe simulatie gedaan waarbij er variabiliteit is toegevoegd. Deze variabiliteit is toegevoegd op alle transistoren in het test circuit en op de weerstands waarde van de geheugen cellen. Voor de transistoren word er een Pelgrom constante voor  $vt$  van 2.5 gebruikt en voor  $\beta$  van 1.2 gebruikt [6]. Voor de weerstand waarde van de memristors word er een gausische verdeling gebruikt met nominale waardes  $7.5\text{k}\Omega$  en  $32.5\text{k}\Omega$  en met  $\sigma = 0.833\text{k}\Omega$ . Er worden telkens 500 monte carlo simulaties gedaan per oplossing. Hierna worden de bitlijn voltages van cellen met een HRS en LRS gefit op een gausische distributie. De oplossing met het grootste bitlijn voltage verschil tussen de extrema van HRS en LRS is een biasload met een switch transistor breete van 100nm, een bias transistor breete van 180nm en een bias voltage van 0V. De bitlijn voltage distributie zijn geillustreerd op figuur 4.4. Het voltage verschil tussen de CDF = 0.1% van HRS en CDF = 99.9% van de LRS is 65mV. Dit is niet veel aangezien de distributie van het bitlijn voltage van de referentie cell hier tussen moet passen en er daarna nog marge over moet zijn voor variabiliteit in de

#### 4. LASTIMPEDANTIE-ANALYSE

---

senseamplifier. Het aandeel to variabiliteit van beide transistoren in de last is even groot.

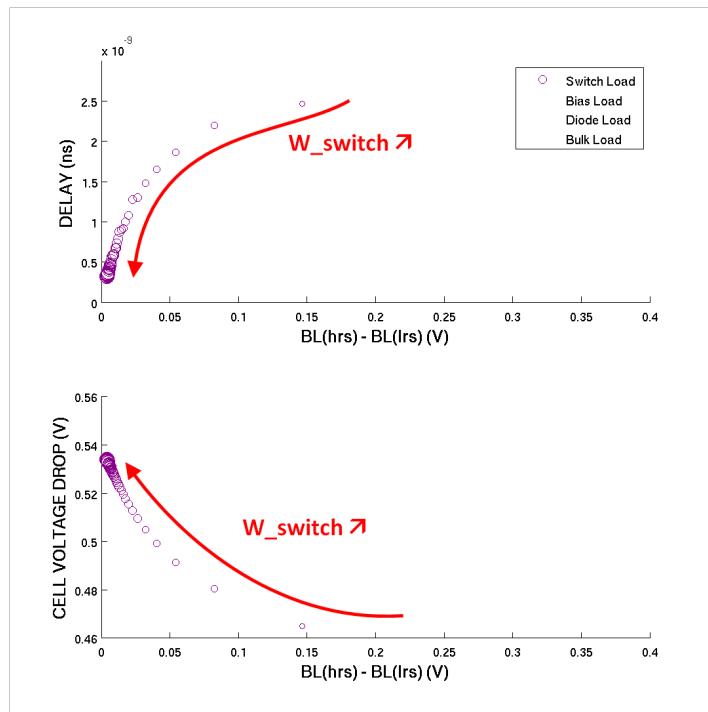


Figuur 4.4: Bitlijn voltage distributie voor een biasload

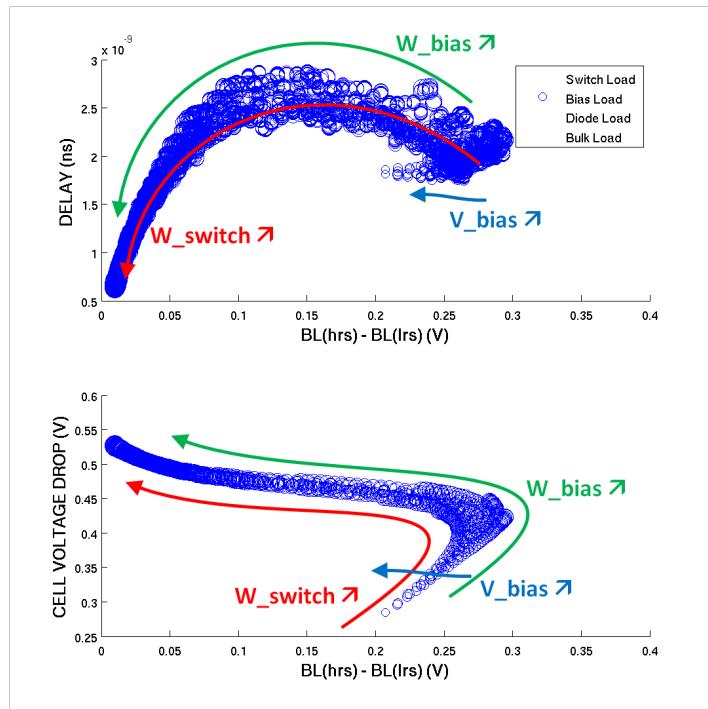
Figuur 4.9 stelt de distributie van het bitlijn voltage van de referentie cellen voor. Hier bij varieert het aantal referentie cellen van 2 tot 30 en er werd een even groot aantal referentie cellen in HRS als LRS gehouden. Zoals gezien kan worden dat met een heel aantal cellen nodig heeft om een distributie breedte van 39mV te krijgen. Dit geeft dan een marge van ongeveer 10mV voor de senseamplifier wat helemaal niet veel is. Daarom wordt de constraint waarbij de transistor lengte minimaal gehouden wordt opgeheven in de volgende sectie.

##### 4.3.3 De transistor lengte vergroten

Om de variabiliteit onder controle te houden moeten de transistoren vergroot worden. Twee opties worden hiervoor overwogen. De eerste is het toevoegen van een derde transistor in serie. Om dezelfde last impedantie te bekomen als voor 2 transistoren in serie, moeten alle drie transistoren een grote breedte hebben wat zou betekenen dat ze groter zijn en minder gevoelig voor mismatch. Een aspect waar niet mee rekening wordt gehouden in die redenering is de toestand waarin deze transistoren zich bevinden. Bij drie transistoren in serie zal de onderste van de drie zich in near tot sub-theashold bevinden. De stroom in het sub-threshold gebied is exponentieel met de gate-source spanning dit leverde grote variatie in de stroom voor kleine vt mismatch. Dit fenomeen zien men niet bij 2 transistoren in serie, aangezien de transistoren hier in het lineare gebied zijn. Daarom wordt gekozen voor een tweede optie om de mismatch onder controle te houden namelijk het vergroten van de lengte van de transistor. Als men de lengte vergroot, stijgt de weerstand wat dan weer gecompenseerd kan worden door de breedte ook wat te vergroten. Nu men deze constraint laten varen heeft, wordt er gezocht om een switchload ipv een



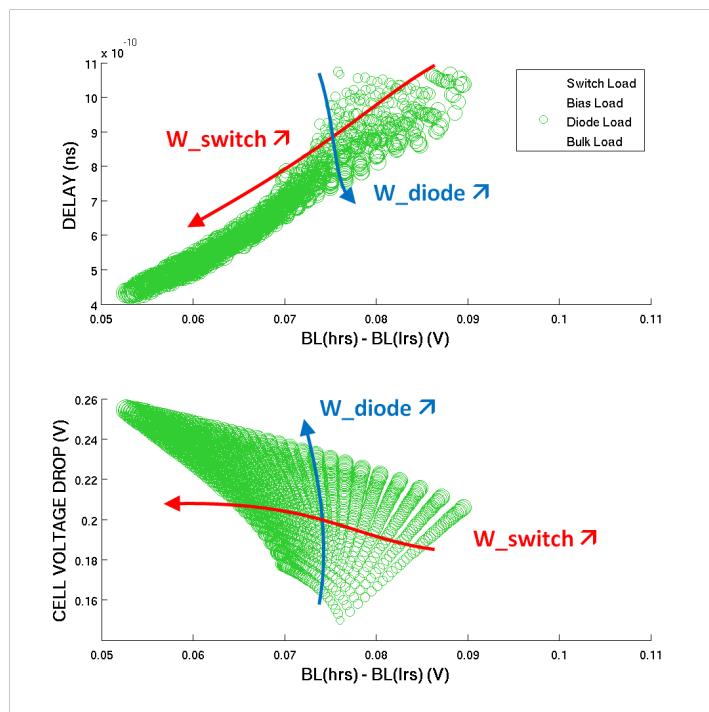
Figuur 4.5: Lineaire sweep van switchload



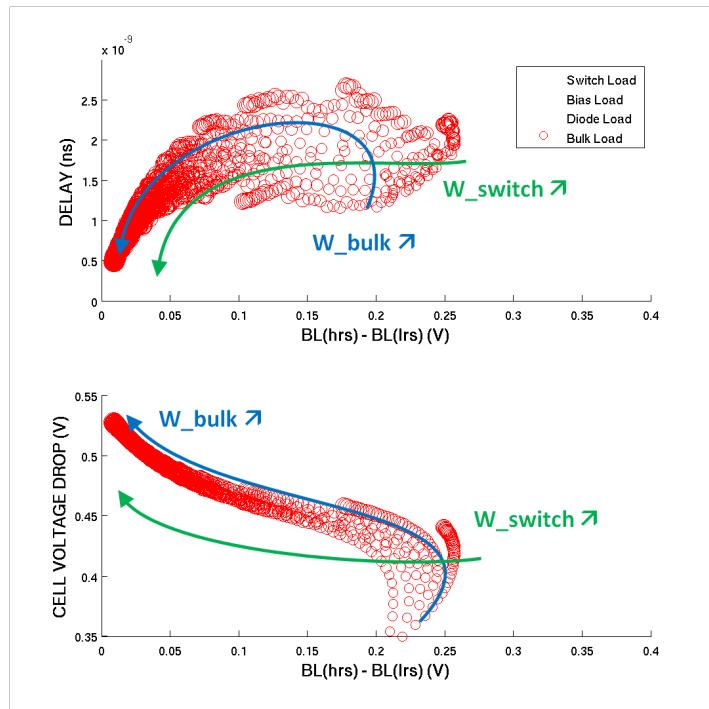
Figuur 4.6: Lineaire sweep van biasload

#### 4. LASTIMPEDANTIE-ANALYSE

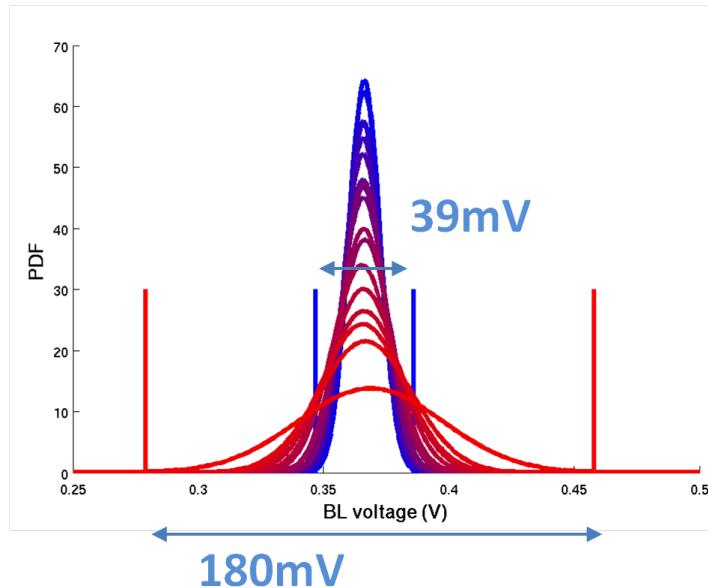
---



Figuur 4.7: Lineaire sweep van diodeload



Figuur 4.8: Lineaire sweep van bulkload



Figuur 4.9: Lineaire sweep van switchload

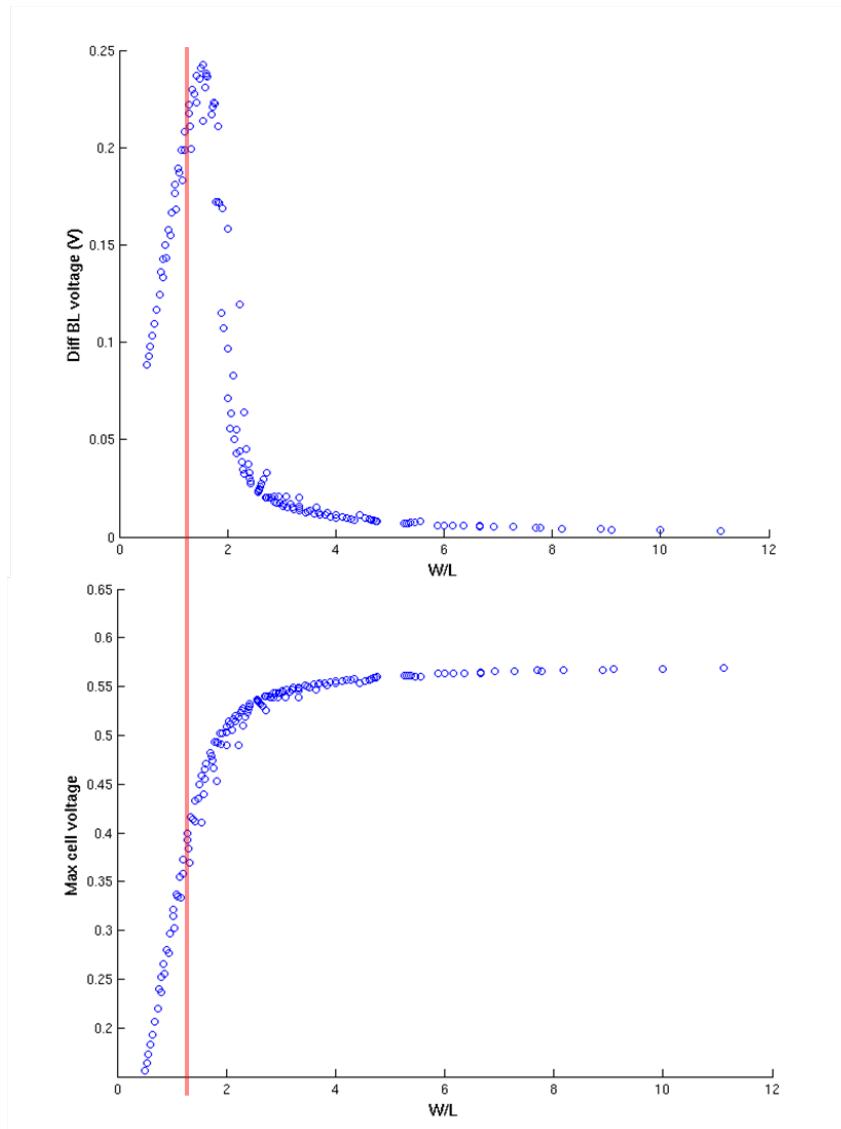
bias load te gebruiken.

Figuur 4.10 geeft de resultaten weer van een sweep van verschillende lengtes en breedtes voor een switchload. De resultaten worden voor gesteld in functie van  $W/L$  wat een indicatie is voor de weerstand van de transistor. In de bovenste figuur kan men duidelijk een maximum zien voor het verschil in bitlijn voltage zoals in sectie 4.1 werd voorspelt. Verder wordt opgemerkt dat er best één van de oplossingen aan de linkerkant van het maximum gekozen wordt aangezien de spannings val over de cell van de oplossingen aan de rechterkant van het maximum te hoog zijn.

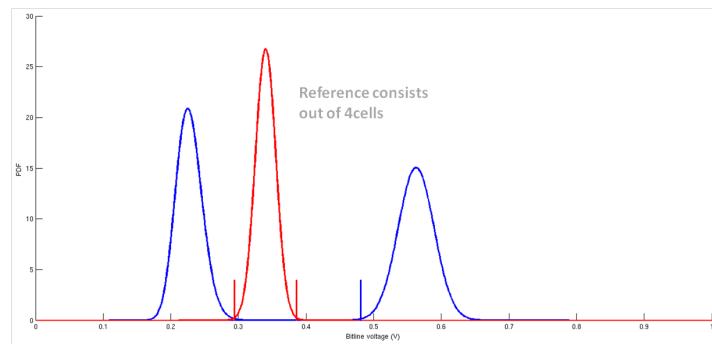
Voor de finale last wordt er geopteert voor een transistor met lengte gelijk aan 198nm en breedte gelijk aan 300nm. Op figuur 4.10 word deze aangeduid met de rode lijn. Op figuur 4.11 word de bitlijn voltage distributie van deze last getoont. Het minimale verschil in bitlijn voltage is bijna 200mV. De distributie van bitlijn voltage van de referentie is ook aangegeven op deze figuur. Deze bestaat hier uit 4 referentie cellen waarvan 2 in HRS en 2 in LRS. Opvallend is dat deze referentie niet in het centrum zit tussen de bitlijn voltages van de cellen. Dit kan opgelost worden door een niet gelijk aantal referentie cellen in HRS en LRS te hebben. Aangezien de standaard deviatie op de bitlijn voltages heel wat beter is nu de lengte van de transistoren ook word gesized, kan er gerust gekozen worden voor een last met een kleiner nominaal verschil in bitlijn voltages. Dit kan 2 voordelen met zich mee brengen. Het eerste is dat de spannings val over de cell verlaagt kan worden als met voor een oplossing kiest dan meer links zit van het maximum in figuur 4.10. Het Tweede is dat men voor een oplossing kan kiezen waarbij de bitlijn voltages lager zijn wat een energie winst kan opleveren. Ondanks deze voordelen werd er toch geopteert voor de oplossing met het grootste bitlijn voltage verschil.

#### 4. LASTIMPEDANTIE-ANALYSE

---



Figuur 4.10: Verschillende oplossingen voor de switchload met variabele lengtes en breedtes



Figuur 4.11: Bitlijn voltage distributie voor de finale load



# Hoofdstuk 5

## Sense Amplifier analyse

Een sense amplifier versterkt kleine signaalverschillen tot rail-tot-rail signalen. Aangezien de uitgangsignalen hiervan ook de uitgelezen bits zijn van het geheugen, is het bovenal belangrijk dat dit op een correcte manier gebeurt, ondanks variabiliteit. Het is dus logisch om de sense amplifier wat meer te onderzoeken en zodanig te ontwerpen op een robuuste manier, terwijl er ook rekening gehouden wordt met energie en snelheid.

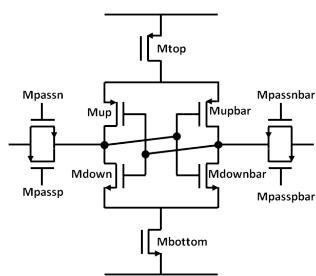
### 5.1 Types SA

...

In wat volgt zal er worden voortgewerkt met de SA van figuur 5.1.

### 5.2 Offsetspanning

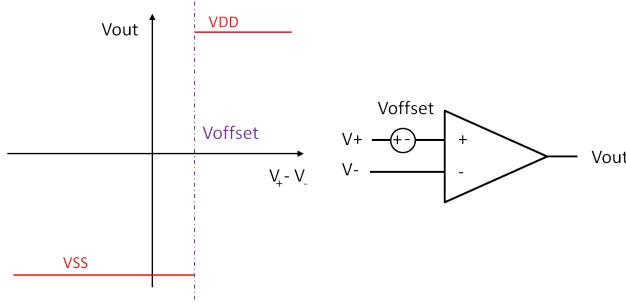
Een ideale sense amplifier zal voor elke twee ingangssignalen correct versterken, tenzij de signalen dezelfde zijn, waarna de SA in een metastabiele toestand belandt. In de praktijk is er echter wegens variabiliteit een limiet voor het ingangsspanningsverschil waarbij er correct versterkt wordt. Deze limiet heet de offsetspanning en wordt geïllustreerd in figuur 5.2. De offsetspanning van een SA is in de ontwerpfasen een stochastische variabele met gemiddelde 0V, pas nadat een chip gefabriceerd is ligt de



Figuur 5.1: een sense amplifier

## 5. SENSE AMPLIFIER ANALYSE

---



Figuur 5.2: Illustratie van offsetspanning

offsetspanning definitief vast [al kan het zijn dat deze met de tijd nog verandert]. Er zijn 2 manieren waarop men de offsetspanning van een systeem kan aanpakken: ofwel ontwerp je het systeem zodanig dat het verschil van de ingangssignalen van de SA groot genoeg is zodat ze [in 99,9% van de gevallen] niet groter is dan de offsetspanning, ofwel bouw je een mechanisme in waarbij je na fabricatie de offsetspanning meet en vervolgens compenseert. In dit werk is gekozen voor het eerste. Hiervoor is het wel belangrijk te onderzoeken wat de verdeling is van de offsetspanning, dit wordt gedaan in de volgende sectie.

### 5.3 Sensitiviteitsanalyse

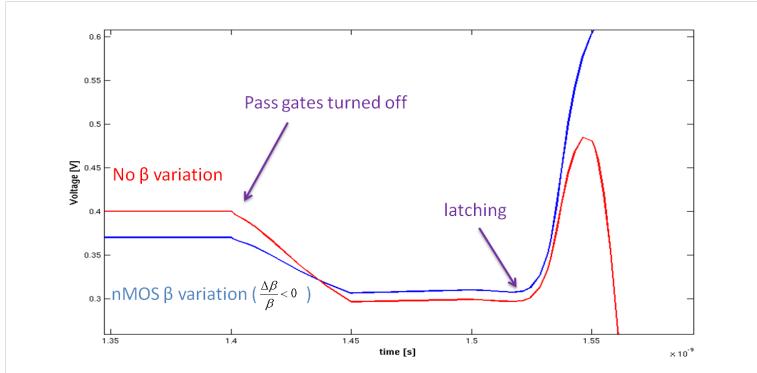
De SA wordt gerealiseerd als een circuit met transistors. Elke transistor heeft 2 stochastische parameters met een normale verdeling, nl.  $\Delta V_t$  en  $\Delta \beta$ . De spreiding van deze verdelingen is bekend:  $\sigma_{\Delta V_t} = \frac{A_{V_t}}{\sqrt{WL}}$  en  $\sigma_{\Delta \beta} = \frac{A_\beta}{\sqrt{WL}}$ . Met een sensitiviteitsanalyse kan men uit deze standaardafwijkingen de standaardafwijking van de offsetspanning  $\sigma_{V_{offset}}$  berekenen. Hierbij wordt verondersteld dat de stochastische variabele  $V_{offset}$  een lineaire combinatie is van de normaal verdeelde afwijkingen  $(\Delta V_t)_i$  en  $(\frac{\Delta \beta}{\beta})_i$ :

$$V_{offset} = \sum_{i=1}^N a_i (\Delta V_t)_i + b_i \left(\frac{\Delta \beta}{\beta}\right)_i$$

a<sub>i</sub> en b<sub>i</sub> zijn de gevoeligheden van de offset naar de variatieparameters:  $a_i = \frac{\partial V_{offset}}{\partial (\Delta V_t)_i}$  en  $b_i = \frac{\partial V_{offset}}{\partial \left(\frac{\Delta \beta}{\beta}\right)_i}$ . Voor een dergelijke variabele geldt

$$\text{dan: } \sigma_{V_{offset}} = \sqrt{\sum_{i=1}^N a_i^2 (\sigma_{\Delta V_t})_i^2 + b_i^2 (\sigma_{\frac{\Delta \beta}{\beta}})_i^2}$$

Er moet wel geverifieerd worden of de stelling dat er een lineaire afhankelijkheid is tussen  $V_{offset}$  en de variatieparameters gegronde is. Dit kan gedaan worden aan de hand van een analyse waarbij elke variatieparameter afzonderlijk geswept wordt. In figuur ?? wordt het resultaat getoond voor een dergelijke analyse bij een SA met minimale afmetingen, in tabel ?? worden de resultaten en de resulterende standaardvariatie van de SA getoond. Er moet opgemerkt worden dat er bij deze simulatie slechts geswept werd voor de variatieparameters van  $-4\sigma$  tot  $4\sigma$ . Dit is om de reden dat voor de minimale transistoren de standaardvariatie het grootst is.



Figuur 5.3: Door  $\beta$ -mismatch is ladingsinjectie van de pass-gates niet meer gematched en gaat de SA foutief latchen

In de Spectre-simulaties zouden transistoren voor te grote negatieve  $\beta$ -mismatch stroom leveren in de omgekeerde richting. Deze situatie zal fysisch nooit optreden.

Opmerkelijk bij deze analyse is dat er een significante bijdrage is van de pass-gates door  $\beta$ -mismatch. Een nadere observatie leert dat deze bijdrage optreedt door ladingsinjectie van de pass-gates die niet meer gematched is (zie figuur 5.3). Hierbij moet wel worden opgemerkt dat voor deze simulatie er geen overlap is tussen het controlesignaal op de passgate aan te zetten en het signaal om de SA te activeren. De reden hierachter is dat als er overlap tussen deze signalen is, de SA ook de BL zou trachten op te laden. Hierbij zou er moeten ingeboet worden aan snelheid en het zou ook extra energie kosten.

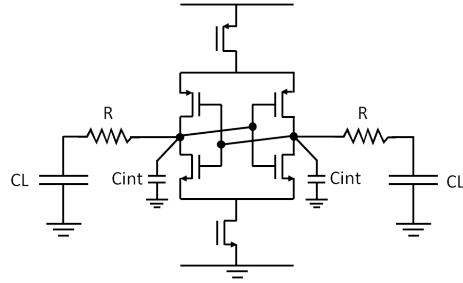
Men kan argumenteren dat er een korte overlap zou kunnen toegelaten zijn, waarna er voldoende spanningsverschil tussen de 2 ingangs-uitgangsknopen zou opgebouwd zijn opdat de ladingsinjectie geen effect meer kan hebben op het eindresultaat. Een tegenargument is dat de timing hiervoor te precies moet zijn.

### 5.3.1 RC-latch-effect

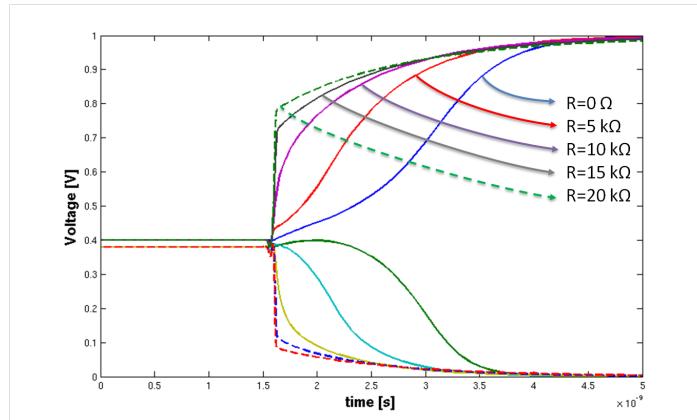
De situatie waarbij er volledige overlap is tussen de controle signalen kan vereenvoudigd worden opgesteld met de situatie van figuur 5.4. De pass-gate die aanstaat wordt voorgesteld als een weerstand, de pass-gate in het local block en diens parasitaire capaciteit wordt verwaarloosd. CL bedraagt voor deze simulatie 46 fF, het equivalent voor een BL waaraan 256 cellen hangen. Cint bedraagt voor een SA met minimale transistorafmetingen 161 aF. Wanneer het dynamisch latch gedrag bekijken wordt voor verschillende waarden van R, treedt er een merkwaardig effect op (zie figuur 5.5): voor voldoende grote waarden van R lijkt het alsof de grote capaciteit ontkoppeld is van de latch tot op een zeker tijdstip, waarna een veel tragere settling optreedt. De verklaring ligt in het feit dat CL zich voor hoge frequenties als een kortsluiting gedraagt (zie figuur 5.6), een plotse stroom vloeit door de weerstand en hierdoor ontstaat er een spanningsval over de weerstand. Hierna gaat er op veel lagere frequenties een spanning beginnen vormen over de capaciteit waardoor de

## 5. SENSE AMPLIFIER ANALYSE

---



Figuur 5.4: Simulatieopstelling voor het RC-latch-effect

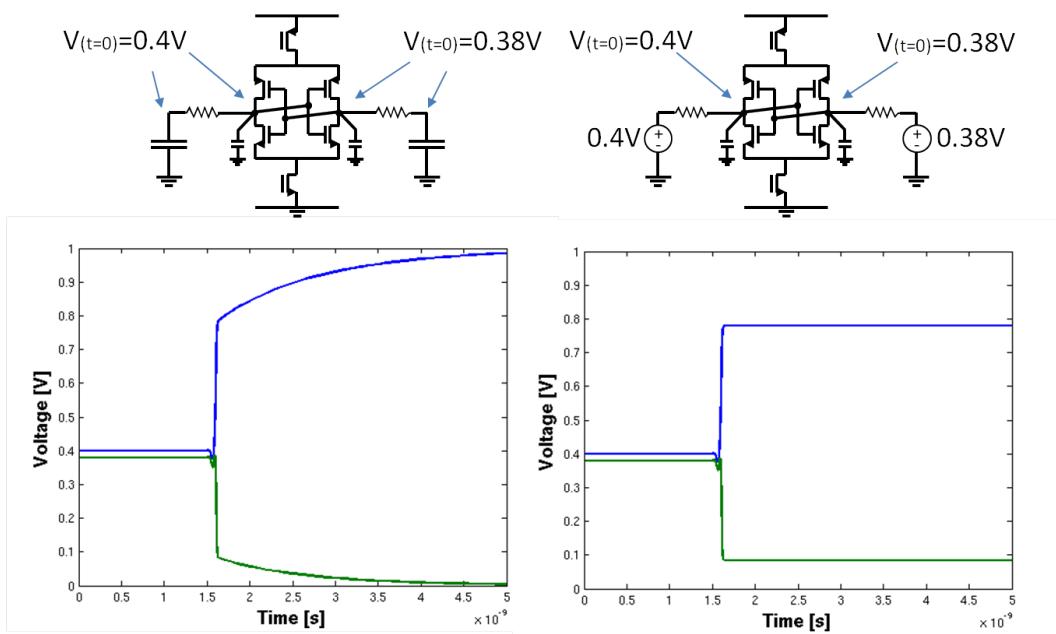


Figuur 5.5: Simulatieresultaten voor het RC-latch-effect: de 2 ingangs-uitgangsknopen zijn voorgeladen op 400mV en 380mV. Na 1,6ns wordt de SA aangezet. De SA is ideaal voor deze simulatie.

ingangs-uitgansknopen volledig kunnen laden/ontladen tot VDD en VSS.

### 5.4 Pareto-simulatie

### 5.5 Besluit



Figuur 5.6: Vergelijking situatie met voorgeladen (eindige) capaciteit en situatie met spanningsbron (oneindige capaciteit)



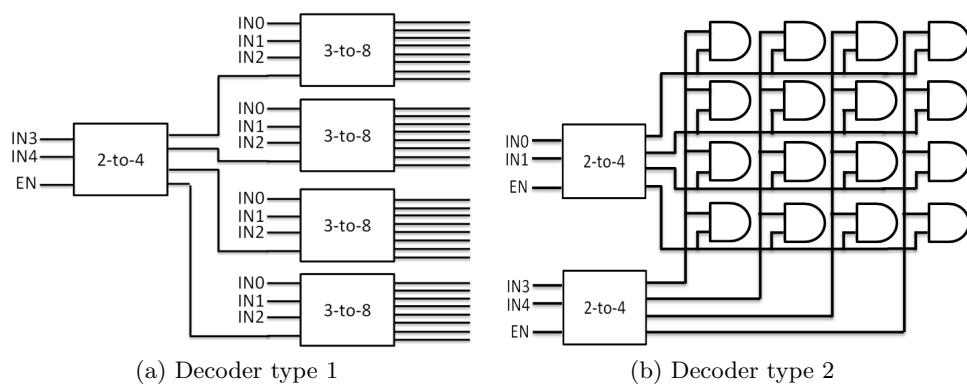
# Hoofdstuk 6

## Omringende logica

Een heleboel logische blokken, zoals decoders, drivers, pass-gates en buffers zitten verwerkt in de geheugenstructuur. In dit hoofdstuk worden deze componenten van wat dichterbij onderzocht.

### 6.1 Decoders

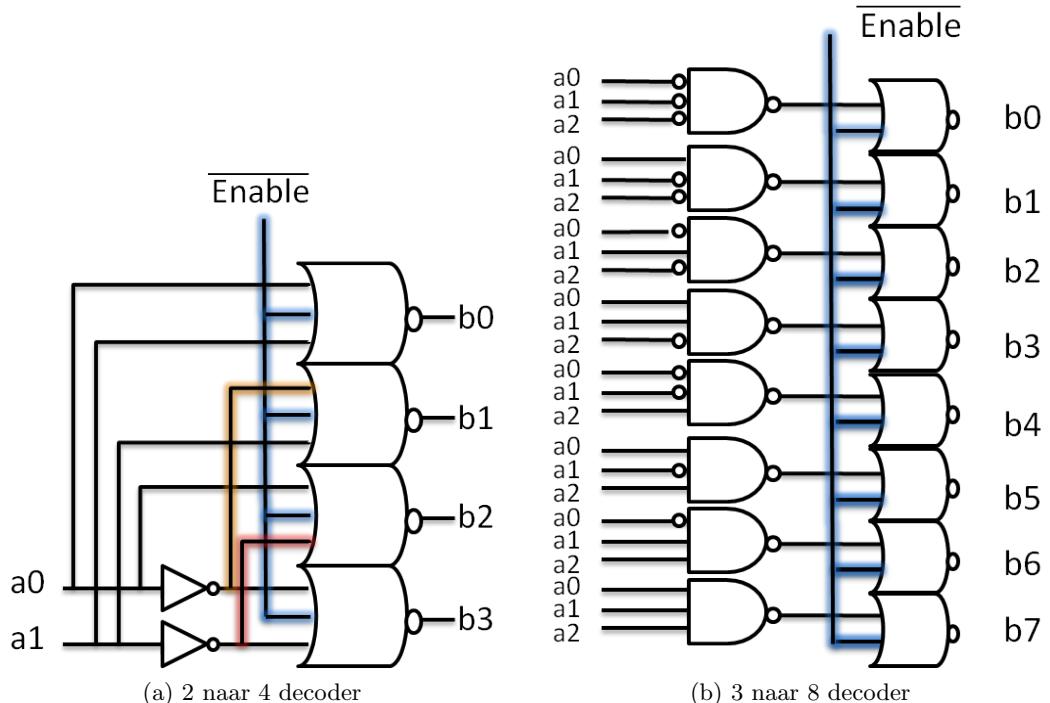
Een decoder is een logische blok dat een uitgang selecteert op basis van een geenco-deerde bus van ingangen. Aangezien het aantal globalblocks, woordlijnen en bitlijnen nog niet vastgelegd werd, werd er een gamma van decoders ontworpen gaande van een 2 naar 4 decoder tot en met een 9 naar 512 decoder. Voor het ontwerp van de grotere decoders, werd er gebruik gemaakt van de kleinere decoders als bouw blokken. Dit kan gedaan worden op 2 manieren; volgens een tree patroon ([6.1a](#)) of volgens een grid patroon ([6.1b](#)). In de volgende secties word het ontwerp van beide manieren toegelicht, en vergeleken.



Figuur 6.1: Opbouw voor grotere decoders

### 6.1.1 De tree decoder

De tree decoder is een decoder met een meerlaagse structuur dat zich uitwaaierd naar de uitgangen. De basis blokken van deze decoder zijn een 2 naar 4 decoder (figuur 6.2a) en een 3 naar 8 decoder (figuur 6.2b). In elke laag in deze structuur komen een aantal ingangen binnen en deze gaan naar de ingang van evenveel decoders als er uitgang zijn in de vorige laag. Elke uitgange van de vorige laag stuurt de enable van een van de decoders van de huidige laag aan. Dit wordt geïllustreerd in de vorm van een 5 naar 32 decoder in figuur 6.1a.



Figuur 6.2: basis decoders

### 6.1.2 De grid decoder

De grid decoder heeft een tweelaagse structuur. De eerste laag bestaat uit een aantal 2 naar 4 en/of 3 naar 8 decoders die in parallel staan. De verschillende uitgangen van deze eerste laag worden dan met AND-gates samen gevoed in een tweede laag. Om glitches te voorkomen is het belangrijk dat al de signalen gelijktijdig binnen komen in de AND-gates, vandaar dat de architectuur van 2 naar 4 decoder van figuur 6.2a veranderd werd tot een AND-OR architectuur zoals de architectuur van de 3 naar 8 decoder. Om de fanout tussen de eerste en tweede laag aan te kunnen, worden de AND-gates van de tweede laag geïmplementeerd als OR gates met inverters aan de ingangen. Deze invertors werden dan afhankelijk van de fanout als buffers gesized.

aantal inputs	aantal 2naar4 decoders	aantal 3naar8 decoders	aantal and gates
1	-	-	-
2	-	-	-
3	-	-	-
4	-	-	-
5	-	-	-
6	-	-	-
7	-	-	-
8	-	-	-
9	-	-	-

Tabel 6.1: dfs

Tabel 6.1 geeft tenslotte weer hoe veel basis decoders er in de eerste laag van de grid decoder zitten en hoeveel and gates er in de tweede laag zitten, in functie van het aantal inputs

### 6.1.3 Vergelijkende studie

Eens ontworpen, kunnen de tree en grid decoders met elkaar vergelijken worden. Naast de gebruikelijke oppervlakte, energie en delay worden ook glitches, mismatch en delay verschillen tussen verschillende adressen onderzocht.

Zoals in figuur 6.3a gezien kan worden, scaalt het oppervlakte van de grid decoder veel minder als die van de tree decoder bij een groter aantal inputs. De plotse stijging in het oppervlakke van de tree decoder met 8 inputs kan verklaart worden door het gebruik van een extra laag in de boom structuur.

Het energie verbruik wordt vergeleken in figuur 6.3b. De grid decoder heeft een lichte stijging van het energie verbruik infunctie van het aantal inputs, dit kan verklaart worden door het aantal gates dat geswitched wordt maar licht stijgt met het aantal inputs, daar naast gaat het meerste van de energie naar de buffers die de tweede laag aansturen (zie figuur 6.4). De tree decoder aan de andere hand heeft een sterker stijging van energie verbruik. dit kan verklaart worden door dat alle decoders in de tree architectuur gedeeltelijk switchen. Dit zou vermindered kunnen worden door de architectuur van de basis decoders (figuur 6.1) te veranderen zodat de enable vooraan komt te staan.

De delay van de decoders kan ook afgelezen worden in figuur 6.3b. Beide decoders hebben ongeveer dezelfde delay. bij grote grid decoders kan de extra delay verklaart worden door het aansturen van een grote fanout.

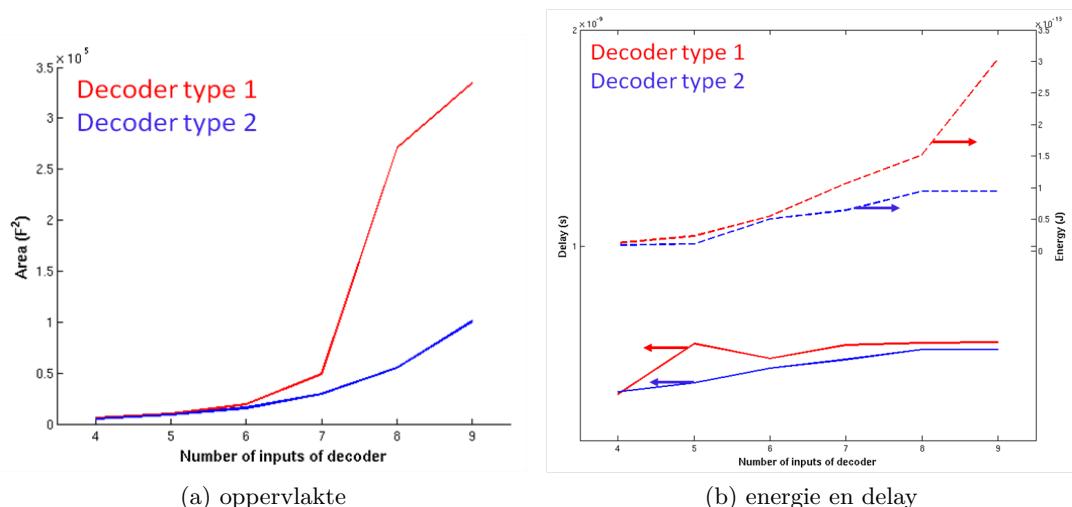
Verder werden het opduiken van glitches onderzocht. In beide decoders is er de mogelijkheid dat er glitches op duiken. Het probleem bij deze decoders is het gebruik van de NOR-gates, de glitch duikt op als de ingang veranderd van een 01 naar een 10 (zie figuur 6.5) en er is een vertraging bij een van de twee ingangen. Bij de tree decoder duiken deze glitches ingebaken in de architectuur aangezien

## 6. OMRINGENDE LOGICA

de enable van een stage aangestuurt wordt door de vorige stage en hier altijd een zekere vertraging is. Bij de grid decoder daarentegen kan er glitch opduiken als de buffers die de tweede laag aansturen een asymetrische delay hebben. Dit kan voor komen bij bv een 5 naar 32 decoder. de uitgangen van de 2naar 4 decoder en de 3 naar 8 decoder die hier in zitten zien een andere last. Deze buffers werden met zorg ontworpen om een asymmetrische delay te voor komen.

Na een snelle mismatch analyse bleek dat de grid decoder minder variatie toon in energie verbruik en delay als de tree decoder. Tenslotte heeft de tree decoder grotere verschillen in delay, afhankelijk van welk het vorige en huidige address van de decoder is. Dit ziet men minder in de grid decoder.

Na het vergelijken van beide decoders op het vlak van oppervlakte, energie, delay, glitches, mismatch en delay verschil, Komt de grid decoder er als het beste uit en deze zal dan ook gebruikt worden in het finale ontwerp.



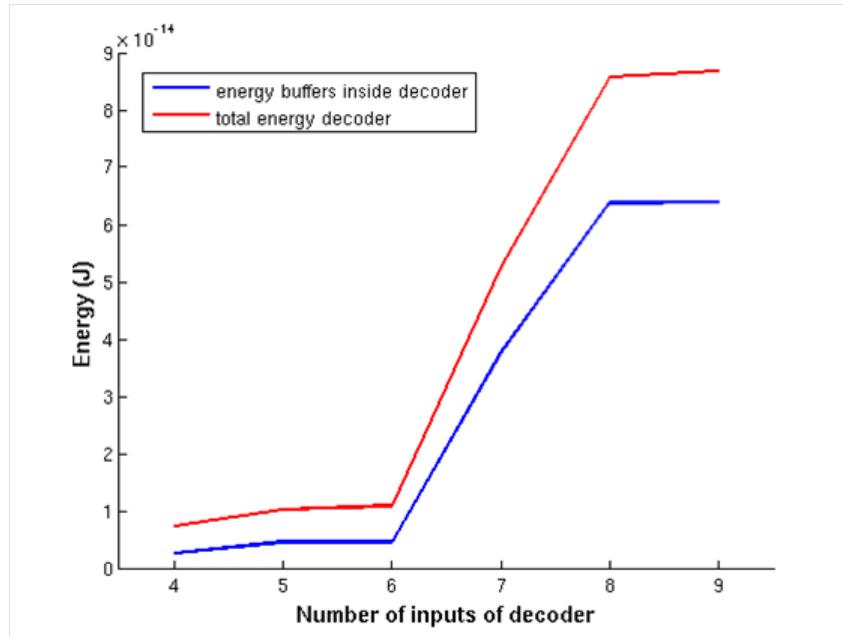
Figuur 6.3: vergelijking van decoder types

## 6.2 Buffers

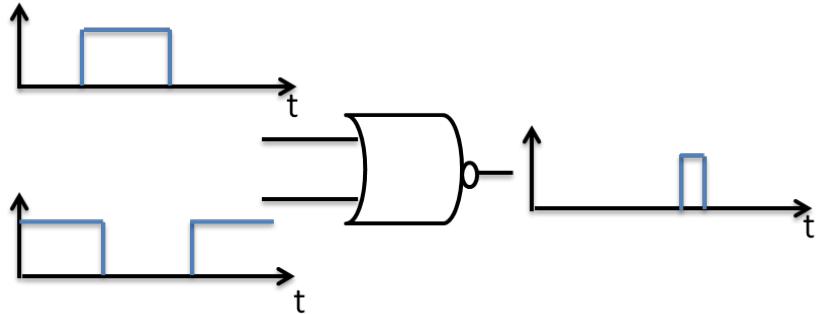
Een digitale buffer is een logische component dat de uitgang van een andere logische block meer drijf kracht geeft om een grotere last sneller aan te sturen. Buffers worden op drie plaatsen in de geheugen architectuur gebruikt. Ten eerste om de woordlijnen aan te sturen. Ten tweede om de referentie logica aan te sturen en ten slotte tussen de eerste en tweede laag in de grid decoders. Tabel 6.2 geeft een overzicht van het type last en het aantal lasten dat de verschillende buffers moeten aansturen.

De buffers werden ontworpen met logical effort waarbij het aantal stages en de sizing van elke stage werd bepaalt volgens het volgende stappen plan:

1. Bepaal de Path effort  $F = GH$  waarbij  $G = 1$  aangezien we enkel met inverters werken en  $H = \frac{C_{out}}{C_{in}}$



Figuur 6.4: Timing globalblock



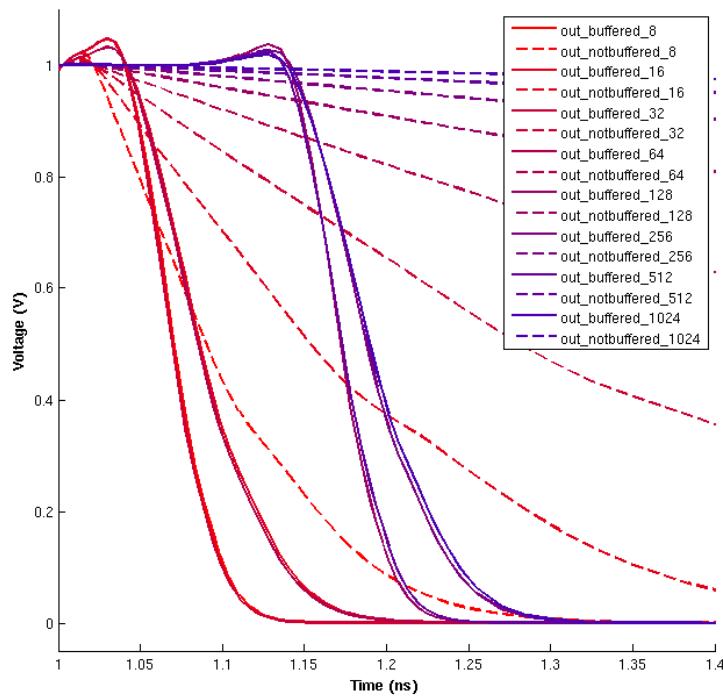
Figuur 6.5: Timing globalblock

2. Het aantal stages wordt bepaalt door  $\hat{N} = \log_4 F$ . hier bij werd er voor een stage effort van 4 gekozen voor een optimale delay [11].  $\hat{N}$  wordt dan afgerond tot een even getal N voor de woordlijn en referentie buffers, en tot een oneven getal voor de decoder buffers.
3. N wordt dan gebruikt om een nieuw stage effort  $\hat{f}$  te berekenen met de formule  $\hat{f} = F^{1/N}$ .
4. Ten slotte kunnen de groottes van de verschillende invertoren in de chain berekend worden met de nieuw stage effort  $\hat{f} = gh$ .

	Last	aantal lasten
Woordlijn Buffer	1 Transistor	#BL
Referentie Buffer	1 Nor + 1 Inv	#BL
Decoder buffer	1 Nor	4 - 64

Tabel 6.2: qsdqsdq

Figuur 6.6 illustreert de ongebufferde en gebufferde signalen die naar de referentie logica gaan, voor een verschillende aantal parallele geplaatste referentie logic.



Figuur 6.6: Timing globalblock

### 6.3 BL- en WL-drivers

### 6.4 Passgates

### 6.5 Besluit

# Hoofdstuk 7

## Timing en optimalisatie

Voor een correcte werking van het geheugen, is het van belang dat de verschillende controlesignalen in een bepaalde volgorde verwerkt en doorgegeven worden. Bovendien is er ruimte voor optimalisatie door al de signalen even snel te maken als het critisch pad. In het eerste deel van dit hoofdstuk zal de invloed van architecture en sizing onderzocht worden op de timing van de signalen. De constraints en vrijheidsgraden die hier uit volgen zullen dan gebruikt worden in het tweede deel van dit hoofdstuk om een optimale architecture te vinden.

### 7.1 Timing

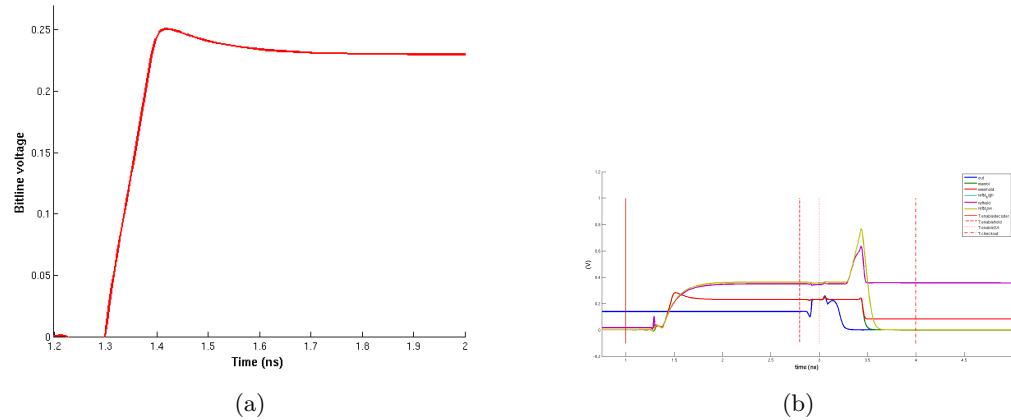
Het ontwerp van dit geheugen gaat tot het niveau van het globalblock 3.1.4, hierbij wordt de veronderstelling gemaakt dat alle signalen tegelijkertijd binnen komem in het globelblock. Hierna propageren de signalen door logica tot ze verschillende transistoren rond de BL aansturen. De aansturing van deze transistoren omvatten de eerste critische timing constraints. Vervolgens worden de muxen en SA aangesloten, deze zullen de tweede timing constraints bevatten.

#### 7.1.1 Critische timing voor het (de)selecteren cell

Timings problemen rond het (de)selecteren van de cell komen door het een verschil in timing voor het (de)selecteren van de load en cell. Indien de load geselecteerd wordt voor de cell zal de bitline vroegtijdig beginnen opladen naar de voedingsspanning. Wanneer de cell dan geselecteert is zal de bitline naar een betekenis volle spanning getrokken worden. Afhankelijk van het tijds verschill tussen deze twee evenementen, zal de bitlijn terug omlaag getrokken worden, wat resulteert in een energie verspilling. Dit wordt geïllustreerd in figuur 7.1a. Indien de cell gedeselecteert wordt voordat de load gedeselecteerd is, Zal de bitline ook opladen naar de voedingsspanning. Dit heeft als gevolg dat het ontladen van de bitlijn langer zal duren en de overbodige oplading resulteert ook in een energie verspilling. Door de keuze van logica (zie figure TODO) zal afhankelijk van het tijds verschill, de mux te vroeg worden afgeschakelt. Waardoor de knoop achter de mux niet volledig ontladen zal zijn. Dit heeft geen nadelige

## 7. TIMING EN OPTIMALISATIE

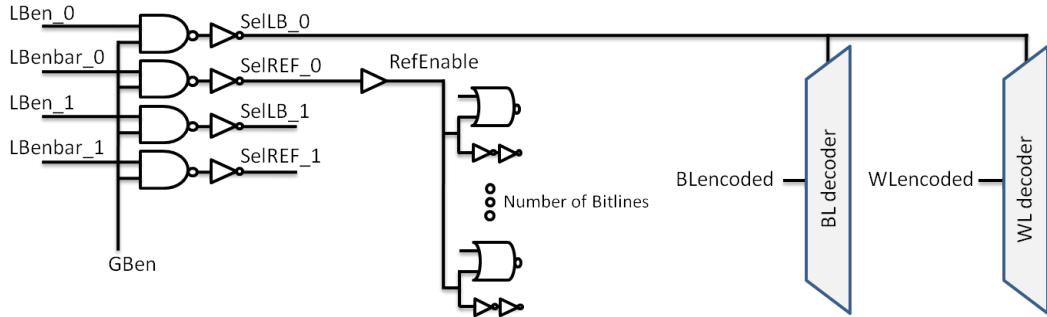
gevolgen door dat de capaciteit op dit knooppunt heel klein is en er bijgevolge een verwaarloosbare ladings injectie is in de volgende lees cyclus. Al dit word geïllustreert in figure 7.1b.



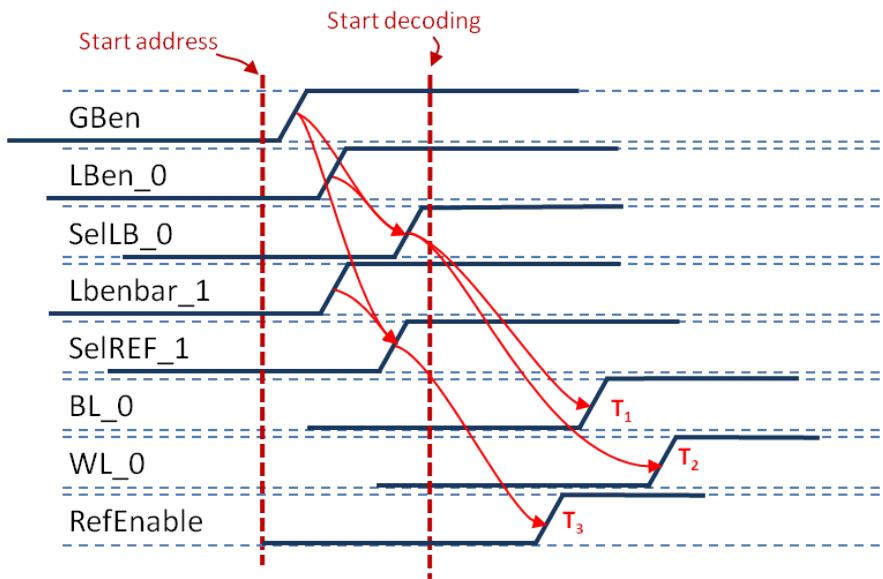
Figuur 7.1: Timing problemen bij de bitlijn

De timing begint in het globalblock. Het circuit en timings diagram word geïllustreerd in figuren 7.2 en 7.3. T1 en T2 stellen het moment voor dat de signalen uit de Bitlijn en Woordlijn decoder komen. T3 stelt het moment voor dat het signaal uit de referentie buffer komt. T1 en T3 zou op hetzelfde moment moeten aankomen om een optimale timing te hebben. Indien dit niet het geval is zal de referentie bitlijnen al aanstaan vooralleer de cell bitlijn aan komt te staan. Indien er een groot aantal referentie bitlijnen zijn zal dit resulteren in een grote energie verspilling. Om deze timing te verwezelen zijn er twee opties. De eerste is het kiezen van een kleine bitlijn decoder en een grote woordlijn decoder. Dit zal voor een kleine T1 zorgen door een kleine delay in de bitlijn decoder. Dit zal een grotere T3 geven door dat de referentie buffer meer capaciteit heeft om op te laden. Een evenwicht kan zo gevonden worden om T1 en T3 op hetzelfde moment te doen verschijnen. Deze eerste optie beperkt de mogelijke architecture intensief en zal timings constraints voor T2 teniet doen zoals later zal blijken. De tweede optie voor het matchen van T1 en T3 is het vertragen van T3. Een vertraging kan gerealiseert worden door het invoeren van delay elementen of door het verslechtern van de buffer. In praktijk is ondervonden dat het invoeren van vertragings element meestal een te grote delay introduceert. Vandaar dat in de final design een buffer wordt gemaakt die niet optimaal is naar snelheid. Om het energie verbruik van de referentie bitlijnen verder te verminderen werden niet al de bitlijnen in de array gebruikt voor het generen van het referentie signaal.

Eens de signalen uit de decoders komen worden deze gevoed in de controle logica voor de memory array. Het circuit en timing diagram is geïllustreerd in figuren 7.4



Figuur 7.2: Globalblock logica



Figuur 7.3: Timing globalblock

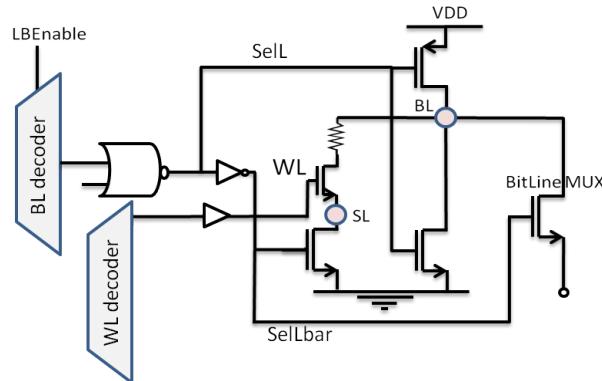
en 7.5. Bij het aanschakelen van de cell zouden de cell vroeger of tegelijk als de last moeten geschakeld worden. Op het timings diagram wordt dit geïllustreert als  $T_4 = T_5 = T_6$ . Door de implementatie van de logica is dit niet mogelijk aangezien er altijd een inverter vertraging verschilt tussen  $T_4$  en  $T_5$ . Deze vertraging is minimaal en kan getollereerd worden door dat de bitlijn in elk geval moet op geladen worden tot minimum  $V_{LRS}$ . Bij lage voedingsspanningen komt dit probleem daarentegen terug boven.  $T_6$  wordt bepaalt door woordlijn decoder en woordlijn buffers. Deze vertraging zou zo gemaakt moeten worden dat deze vroeger of gelijk met  $T_5$  valt. Bij het afschakelijke van de cell zijn de omgekeerde voorwaarden nodig namelijk, De last zou vroeger of gelijk als de cell moeten afgeschakeld moeten worden. Deze voorwaarde is voldaan als  $T_7$  voor  $T_8$  en  $T_9$  komt. Door de inverter is  $T_7$  altijd voor  $T_8$ ,  $T_9$  daarentegen word bepaalt door de woordlijn decoder en buffer en zou voor

## 7. TIMING EN OPTIMALISATIE

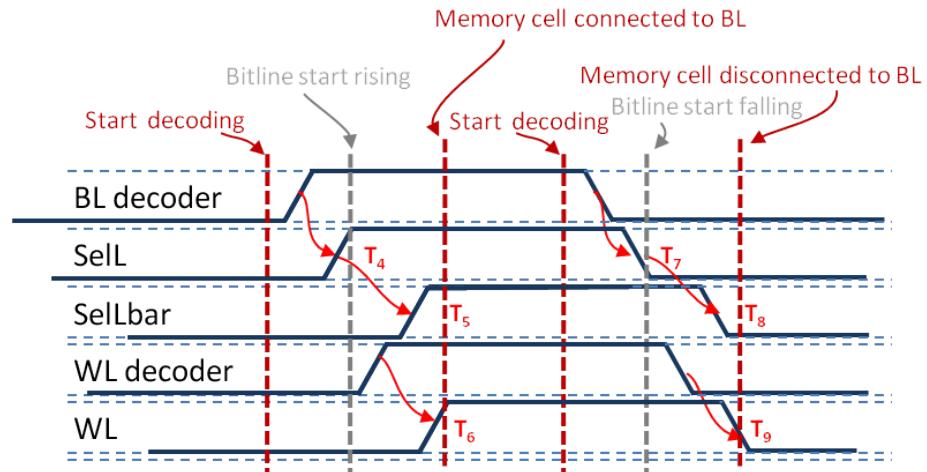
---

T7 moeten komen.

De timing van de controle logica voor de memory array staat in het circuit vast op de timing van de woordlijn na. Deze moet moet geselecteerd worden voor de sourcelijn geselecteerd is en moet gedeselecteerd worden na dat de last gedeselecteerd is. De timing van de woordlijn wordt explicet bepaald door de grote van de woordlijn decoder en impliciet door de grote van de bitlijn decoder dat de grote van de woordlijn buffer bepaalt. Figure 7.6 geeft de delay van verschillende groottes van woordlijn decoders + buffer ifv verschillende groottes van bitlijn decoders weers ... TODO

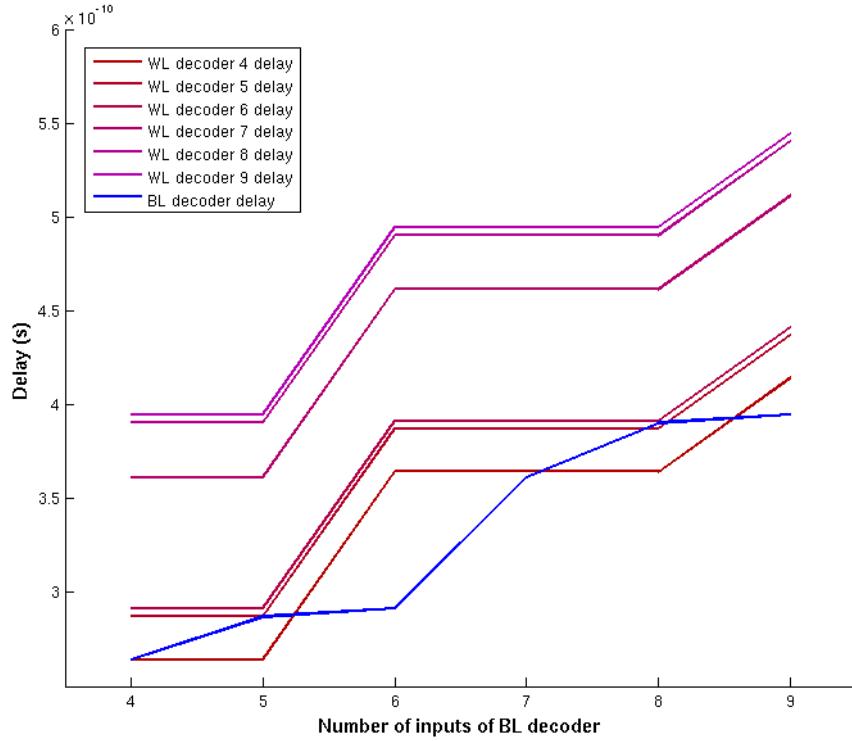


Figuur 7.4: Controle logica memory array



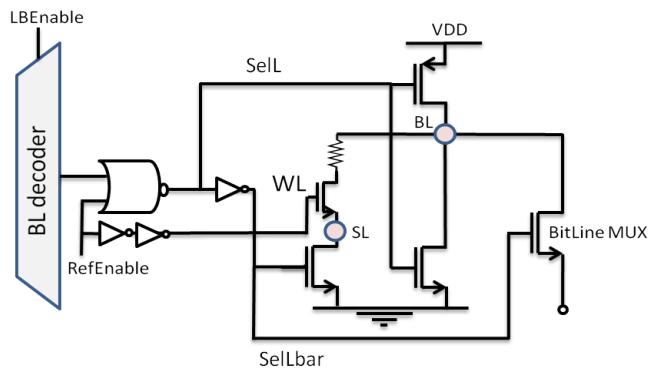
Figuur 7.5: Timing controle logica memory array

De timings voorwaarden voor het selecteren en deselecteren van de referentie cel, zijn hetzelfde als die van de memory cellen. Het circuit en timing diagram is geïllustreerd in figuren 7.7 en 7.8. Anders als bij de memory cellen worden de timings voorwaarden al in de logica zelf voldaan door dat de woordlijnen worden aangestuurd

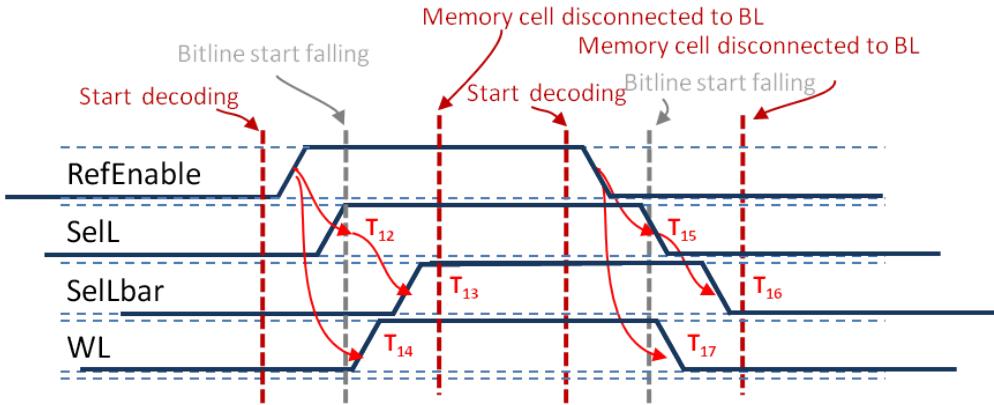


Figuur 7.6: Delay van woordlijn decoders + buffers ifv bitlijn decoders

door een signaal dat rechtstreeks van de bitlijn decoder komt. Dit signaal wordt dan vertraagd door twee invertoren om de juiste timing te verwijzelen.



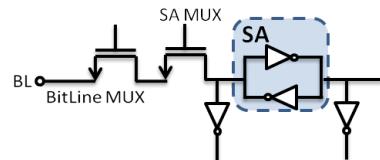
Figuur 7.7: Controle logica memory array



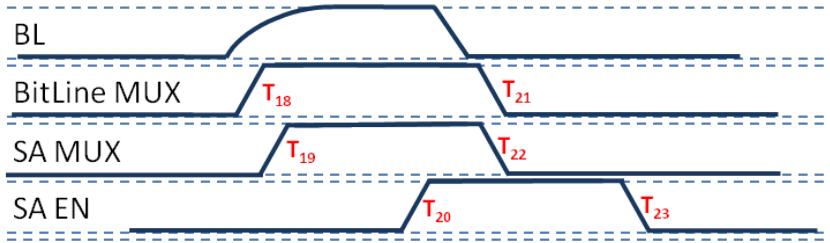
Figuur 7.8: Timing controle logica memory array

### 7.1.2 Critische timing voor het uitlezen van de cell

Eens de cel geselecteerd is wordt de bitlijn opgeladen. De volgende stap is dit signaal voeden aan de sense amplifier. Dit signaal wordt eerst door een eerst mux geleid om uit de localblock te geraken. Vervolgens wordt het signaal door een tweede mux geleid die als sample en hold dient voor de sense amplifier. Figuren 7.9 en 7.10 illustreren het circuit en timing rond de sense amplifier. Eens de bitlijn is aangesloten wordt de eerste mux automatisch aangesloten zoals uitgelegd in de vorige paragrafen. T19 stelt het tijdstip voor waar de tweede mux aangesloten moet worden. Deze timing is niet crutiaal, de mux mag zowel aangesloten worden voor als na het aansluiten van de bitlijn mux. Het tijdstip van afsluiten van deze mux (T22) is daarentegen wel belangrijk. Dit moet namelijk gebeuren voor de bitlijn mux afgesloten is (T21) anders zullen er 2 charge injections voorkomen ipv een. Om een zo snel mogelijke latching van de sense amplifiers te hebben, is het tijd stip waar de sense amplifier (T20) aangesloten word belangrijk. Wanneer de sense amplifier juist word aangesloten treed er een latching effect plaats waar de sense amplifier zich gedraagt als of er geen last aan hangt dit effect werd beschreven in sectie ???. Eens dit voorbij is zal de sense amplifier latchen aan de snelheid van de RC constante van de bitlijn. Om een snelle latching te hebben moet de tweede mux afgesloten worden voor dit latching effect voorbij is. Tenslotte kan de sense amplifier afgesloten worden eens het latchen voorbij is.



Figuur 7.9: logica rond SA



Figuur 7.10: Timing logica rond SA

## 7.2 Analyse verschillende geheugencconfiguraties

Het finale geheugen zal 4 Mbit groot zijn. Heel wat configuraties zijn mogelijk om deze grote te verwezenlijken. Om deze mogelijkheden wat in te perken worden de volgende beperking op gelegd. Het aantal Woordlijnen moet groter of gelijk zijn aan het aantal bitlijnen. Hierdoor zal Het ontladen van de bitlijn sneller verlopen. Dit levert 20 mogelijke configuraties voor aantal bitlijnen,woordlijnen en globalblocks. Deze configuraties worden vergeleken op basis van hun oppervlakte, energie verbruik en leessnelheid.

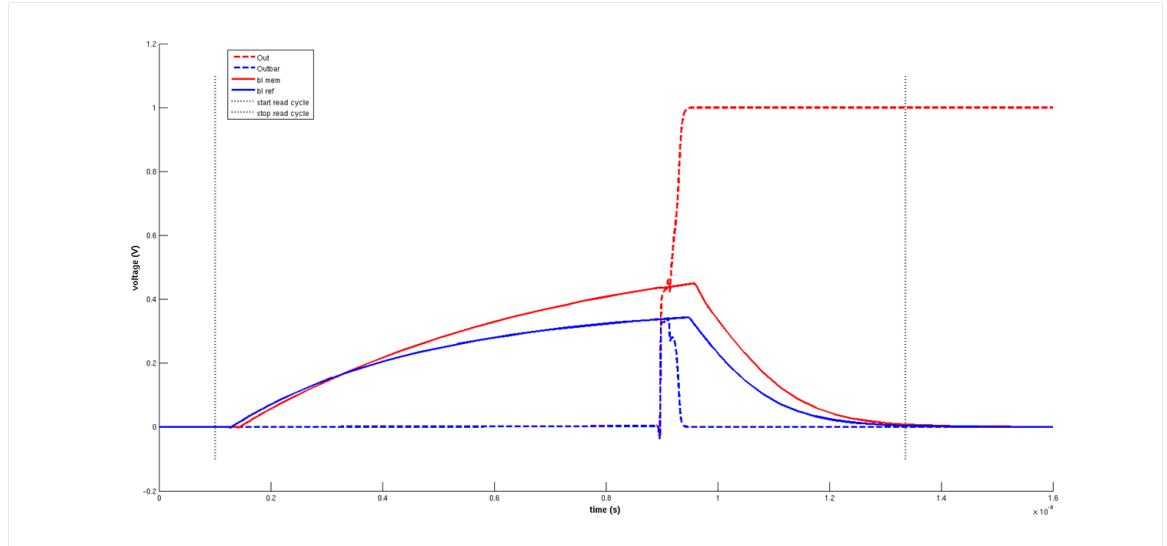
### 7.2.1 Evaluatie criteria voor de geheugencconfiguraties

Het Oppervlakte wordt berekend op basis van de lengtes en breedtes van de transistoren. Gardrings en verbindingen lijnen worden niet meegerekend in de berekeningen van het oppervlakte van de logica. De lengte van de geheugen cellen wordt  $1.5*6F$  genomen en voor de breedte wordt  $2*6F$  genomen [4]. Hoewel deze afmetingen voor een MTJ geheugen cell zijn, geven ze een goede schatting van het oppervlakte van een Memristor geheugencell. Verder is in dit oppervlakte de grote van bitlijn, woordlijn en selectlijn meegekend.

Het energie verbruik word berekend door de stroom van de voedingsspanning te integreren over de tijd en te vermenigvuldigen met de voedingsspanning. De signalen die binnen komen in een global block, komen van ideale spice bronnen. Dit geeft als gevolg dat er een charge injectie is naar de voedingsspanning (zie bijlage A). Dit heeft een invloed op de energie berekeningen maar er worden geen pogingen gedaan op deze te corrigeren. Verder wordt het aantal referentie cellen ook constant gehouden voor de verschillende configuraties. Dit wordt gedaan om het energie verbruik te verkleinen en omdat men maar een beperkt aantal cellen nodig heeft om een goede referentie te krijgen.

De leessnelheid is afhankelijk van de verschillende controle signalen in de lees cyclus en deze gebeurd als volgt. De lees cyclus begint wanneer de signalen binnen komen in de globalblock. De SA wordt aangesloten wanneer het verschil tussen de memory bitlijn en de referentie bitlijn 100mV bedraagt. Hierdoor wordt het tijds verschil tussen geheugens met een klein aantal woordlijnen en geheugens met een groot aantal woordlijnen verkleind, wat een betere concurrentie geeft. In het finale geheugen ontwerp zal de leessnelheid verder opgedreven worden door deze 100mV voltage

verschil te verkleinen. Verder word er ook altijd een cell met een HRS uitgelezen aangezien deze de bitlijn langer moet opladen om tot aan de 100mV verschil te komen wat een realistischere leessnelheid geeft. De lees cyclus eindigt wanneer het bitlijn voltage terug naar de grond is getrokken. Figure 7.11 illustreert de hele leescyclus.



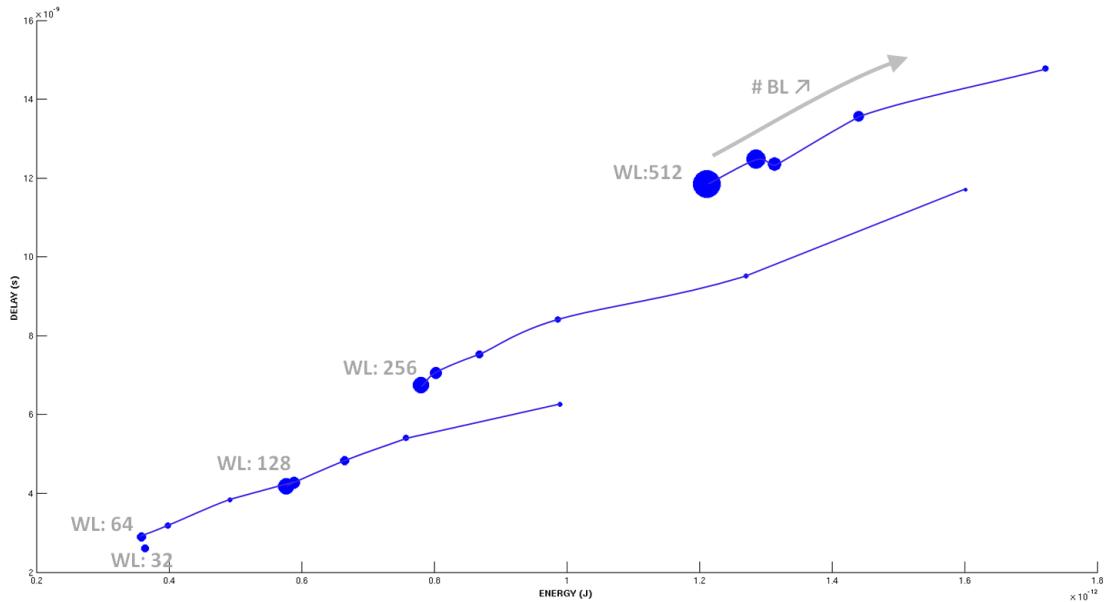
Figuur 7.11: Timing controle logica memory array

### 7.2.2 Vergelijking van de geheugenconfiguraties

Er werden 20 mogelijke geheugenconfiguraties geselecteert als kandidaat voor het finale ontwerp, hun positie in de evaluatie ruimte wordt getoont in figuur 7.13. Hierop staat het energie verbruik op de x-as, de delay op de y-as en het oppervlakte gebruik wordt geïndiceert door de grote van de punten. Het energie verbruik en delay wordt voornamelijk bepaalt door het aantal woordlijnen en bitlijnen. De delay wordt voornamelijk bepaalt door het opladen van de bitlijnen, wat dan ook de voornaamste vorm van energie verbruik is. De snelheid van de bitlijnen wordt dan weer bepaalt door het aantal woordlijnen wat gezien kan worden in figuur 7.12. Het aantal bitlijnen beïnvloed dan weer meer het energie verbruik. Dit extra energie verbruik gaat teneerste naar de woorlijn buffers, tentweede naar de bitlijn decoders en tenslotte naar de bitlijn zelf. Deze laatste is door dat de bitlijn bij het deselecteren van de cell langer aanblijft dan bij een kleiner aantal bitlijnen. Dit is ook de reden waarom een groter aantal bitlijnen een langere delay heeft. Bij alle geheugen configuraties gaat het vermogen verbruik eerst naar de geheugecell, vervolgens naar de logica, vervolgens naar de buffers en ten slotte naar de sense amplifiers. Het oppervlakte wordt bepaalt door het aantal globalblocks en de grote van de decoders. Een groot aantal woordlijnen in combinatie met een klein aantal bitlijnen geeft de noodzaak aan een groot aantal globalblocks en dit geeft een groot oppervlak als gevolg.

Als conclusie kan gezegd worden dat de optimale geheugen configuratie bestaat uit

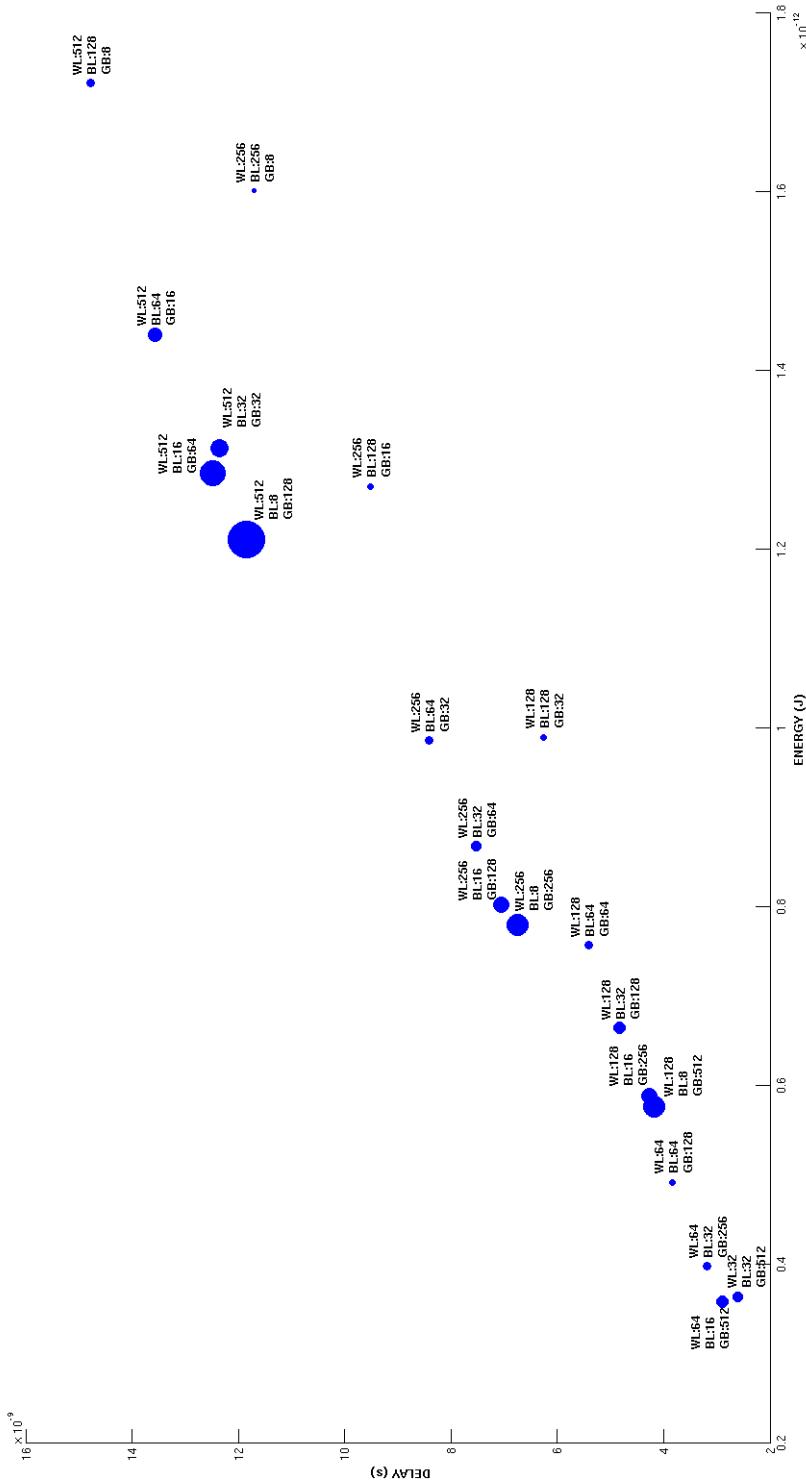
een klein gelijk aantal woordlijn en bitlijnen wat een optimum zal geven voor energie verbruik en delay, en een sub optimum zal geven voor oppervlakte.



Figuur 7.12: Timing controle logica memory array

## 7. TIMING EN OPTIMALIZATIE

---



Figuur 7.13: Timing controle logica memory array

## **Hoofdstuk 8**

### **Besluit**

De masterproeftekst wordt afgesloten met een hoofdstuk waarin alle besluiten nog eens samengevat worden. Dit is ook de plaats voor suggesties naar het verder gebruik van de resultaten, zowel industriële toepassingen als verder onderzoek.



# **Bijlagen**



## Bijlage A

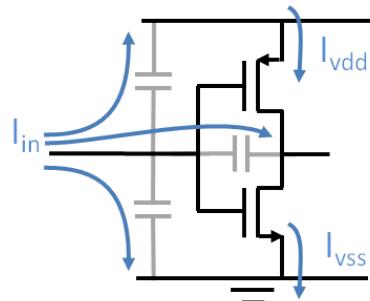
# Charge injectie met ideale spice bronnen.

Bij het berekenen van de energie consumptie van verschillende bouwblokken, valt het op dat de voeding, stroom opneemt ipv afgeeft bij het schakelen van de ideale spice bronnen. Dit komt door een charge injectie van de ideale spice bronnen. Om dit te verificeren, worden de stromen van een simpele inverterschakeling bestudeert. Figuur A.1 stelt een simple inverterschakeling voor met relevante parasite capaciteiten. Bij het plaatsen van een stap functie aan de inverter, zal er een lading door de capaciteiten vloeien. Omdat de positieve stroom die geobserveert wordt in de voeding afkomstig komt van de ingang, moet de som van de stroom door de ingang, voeding en grond moet nul zijn. De stroom door de ingang kan in Spice opgemeten worden door een weerstand met resistieve waarde gelijk aan nul, in serie met de ingang te zetten.

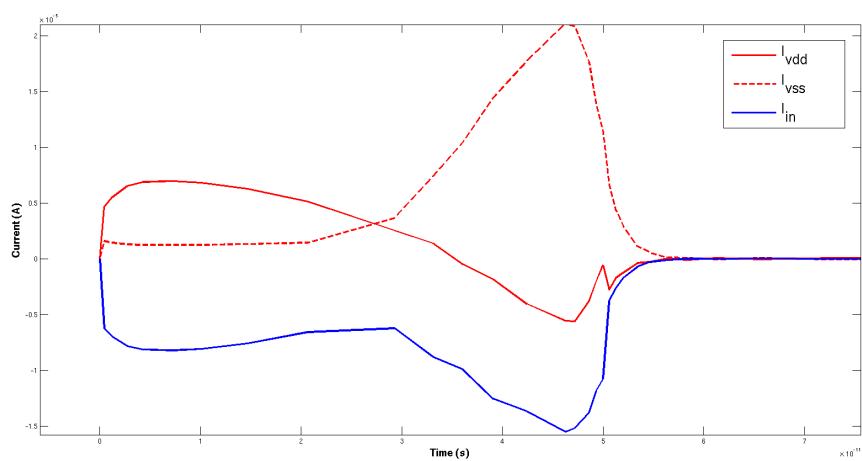
Figure A.2 toont deze drie stromen. In eerste deel van de figuur (tot tijdstip  $3 \cdot 10^{-11}$ ) is de input al aan het stijgen maar de inverter is nog niet aan schakelijk. De voeding en grond stromen komen dan puur van de ingang. Vanaf tijdstip  $3 \cdot 10^{-11}$ , is de inverter aan het schakelen en is er een aandeel van de stroom in de grond dat uit de voeding komt. De som van alle drie stromen is ten alle tijden gelijk aan nul. In conclusie is hierbij aangetoont dat er een charge injectie is van ideale spice bronnen in het circuit. Dit heeft een invloed op de stromen en daardoor ook de energie berekeningen, maar dit verwaarlozen we bij onze berekeningen.

#### A. CHARGE INJECTIE MET IDEALE SPICE BRONNEN.

---



Figuur A.1: Timing globalblock



Figuur A.2: Timing globalblock

# Bibliografie

- [1] I. Baek, M. Lee, S. Seo, M.-J. Lee, D. Seo, D. S. Suh, J. Park, S. Park, T. Kim, I. Yoo, U.-i. Chung, and J. Moon. Highly scalable nonvolatile resistive memory using simple binary oxide driven by asymmetric unipolar voltage pulses. In *Electron Devices Meeting, 2004. IEDM Technical Digest. IEEE International*, pages 587–590, Dec 2004.
- [2] Y.-Y. Chen, L. Goux, L. Pantisano, J. Swerts, C. Adelmann, S. Mertens, V. Afanasiev, X. Wang, B. Govoreanu, R. Degraeve, S. Kubicek, V. Paraschiv, B. Verbrugge, N. Jossart, L. Altimime, M. Jurczak, J. Kittl, G. Groeseneken, and D. Wouters. Fully cmos beol compatible hfo<sub>2</sub> rram cell, with low program current, strong retention and high scalability, using an optimized plasma enhanced atomic layer deposition (peald) process for tin electrode. In *Interconnect Technology Conference and 2011 Materials for Advanced Metallization (ITC/MAM), 2011 IEEE International*, pages 1–3, May 2011.
- [3] L. . CHUA. Memristor-the missing circuit element. *IEEE Transactions of circuit theory*, September 1971.
- [4] S. Cosemans. Intro regarding read sensing schemes. powerpoint presentation, 2013.
- [5] K. M. Kim, B. J. Choi, B. W. Koo, S. Choi, D. S. Jeong, and C. S. Hwang. Resistive switching in pt/al<sub>2</sub>o<sub>3</sub>/tio<sub>2</sub>/ru stacked structures. *Electrochem. Solid State Lett.*, 9G343–G346, 2006.
- [6] K. J. Kuhn. Variation in 45nm and implications for 32nm and beyond. powerpoint presentation, 2009.
- [7] G. E. MOORE. Cramming more components onto integrated circuits. *Electronics*, April 1965.
- [8] K. Prall and K. Parat. 25nm 64gb mlc nand technology and scaling challenges invited paper. In *Electron Devices Meeting (IEDM), 2010 IEEE International*, pages 5.2.1–5.2.4, Dec 2010.
- [9] F. Ren, H. Park, R. Dorrance, Y. Toriyama, C.-K. Yang, and D. Markovic. A body-voltage-sensing-based short pulse reading circuit for spin-torque transfer

## BIBLIOGRAFIE

---

- rams (stt-rams). In *Quality Electronic Design (ISQED), 2012 13th International Symposium on*, pages 275–282, March 2012.
- [10] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams. The missing memristor found. *Nature*, 453(7191):80–83, 2008.
  - [11] I. Sutherland, B. Sproull, and D. Harris. *Logical Effort: Designing Fast CMOS Circuits*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.
  - [12] H. S. P. Wong, H.-Y. Lee, S. Yu, Y.-S. Chen, Y. Wu, P.-S. Chen, B. Lee, F. Chen, and M.-J. Tsai. Metal oxide rram. *Proceedings of the IEEE*, 100(6):1951–1970, June 2012.

## Fiche masterproef

*Studenten:* Wouter Diels  
Alexander Standaert

*Titel:* Ontwerp van een RRAM geheugen

*Engelse titel:* The best master thesis ever

*UDC:* 621.3

*Korte inhoud:*

Hier komt een heel bondig abstract van hooguit 500 woorden. L<sup>A</sup>T<sub>E</sub>X commando's mogen hier gebruikt worden. Blanco lijnen (of het commando \par) zijn wel niet toegelaten!

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Thesis voorgelegd tot het behalen van de graad van Master of Science in de ingenieurswetenschappen: elektrotechniek, optie Elektronica en geïntegreerde schakelingen

*Promotor:* Prof. dr. ir. W. Dehaene

*Assessoren:* Prof. dr. ir. R. Lauwereins  
Prof. dr. ir. M. Verhelst

*Begeleiders:* ir. B. Baran  
dr. ir. S. Cosemans