

# “Low Power Design”

2023-2024

Prof. FRUSTACI

## SRAM Analysis

Date	<10/11/2023>
Document	Final Document

Full Name	ID	E-mail Address
Giorgio Ubbriaco	247284	bbrgrg00h11d086x@studeni.ti.unical.it

# Contents

<b>DEFINITIONS .....</b>	<b>4</b>
DIRECTORIES PROJECT PATHS.....	4
GLOBAL VARIABLES.....	4
GLOBAL FUNCTIONS.....	6
<b>I.    TASKS TO BE PERFORMED .....</b>	<b>7</b>
<b>II.   ABSTRACT .....</b>	<b>8</b>
<b>1.    INTRODUCTION .....</b>	<b>9</b>
1.1.   SRAM 6T .....	9
1.2.   HOLD PHASE .....	9
1.3.   READ PHASE.....	9
1.4.   SNM.....	10
1.4.1. <i>Graphical Method</i> .....	10
1.4.2. <i>Seevinck Method</i> .....	11
1.5.    NORMAL DISTRIBUTION.....	12
1.6.   LOG-NORMAL DISTRIBUTION.....	12
1.7.   VDD SCALING .....	13
1.8.   DRV .....	13
<b>2.    ANALYSIS AND DESIGN OF A 6T SRAM VIA GRAPHICAL METHOD .....</b>	<b>15</b>
2.1.   HOLD PHASE .....	15
2.2.   READ PHASE.....	18
<b>3.    ANALYSIS AND DESIGN OF A 6T SRAM VIA SEEVINCK METHOD .....</b>	<b>21</b>
3.1.   HOLD PHASE .....	21
3.2.   READ PHASE.....	23
3.3.   COMPARATIVE ANALYSIS BETWEEN GRAPHICAL METHOD AND SEEVINCK METHOD .....	25
<b>4.    VDD SCALING OF A 6T SRAM CONSIDERING GAUSSIAN VTH.....</b>	<b>28</b>
4.1.   HOLD PHASE .....	29
4.2.   READ PHASE.....	31
4.3.   I <sub>LEAK</sub> ANALYSIS.....	32
4.4.   CELL ROBUSTNESS AND LEAKAGE POWER.....	33
4.5.   DRV ANALYSIS.....	34



# Definitions

Qui sono contenute diverse definizioni utilizzate all'interno del report per semplificare la fluidità delle analisi.

## Directories Project Paths

- data = "./python/sram-analysis/data"
- Itspice = "./Itspice/models/"
- rit\_models\_for\_Itspice\_file\_path = "./Itspice/utils/utils-montecarlo-experiments/RIT\_Models\_For\_LTSPICE.txt"
- rit\_models\_for\_Itspice\_file\_path = "./Itspice/utils/utils-montecarlo-experiments/RIT\_Models\_For\_LTSPICE\_MonteCarlo.txt"

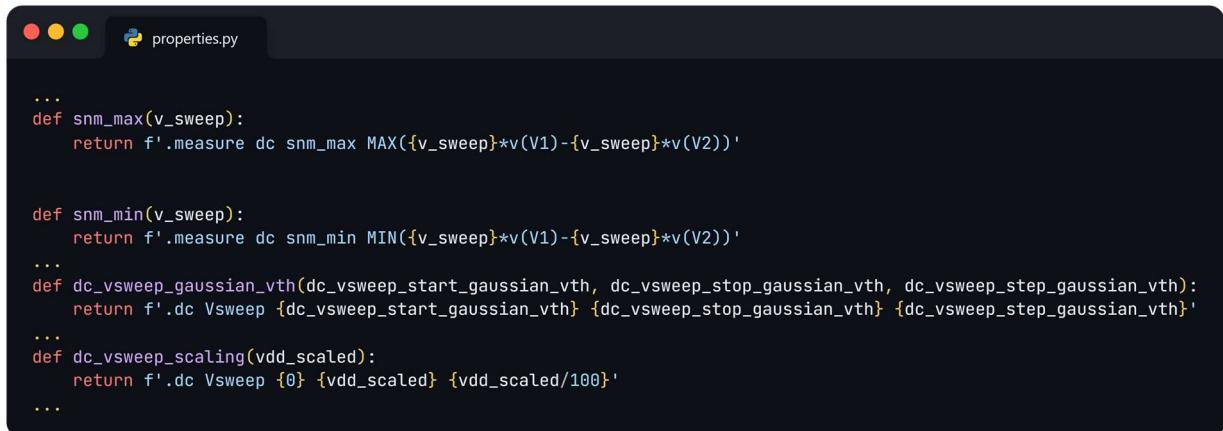
## Global Variables

- w\_ax\_pos = 1
- vwl\_hold = '0'
- vbl\_hold = '1'
- vblneg\_hold = '1'
- vwl\_read = '1'
- vbl\_read = '1'
- vblneg\_read = '1'
- w\_ax\_start = 0.12
- w\_ax\_stop = 0.24
- iterations\_scaling = 20
- vdd\_start = 1.0
- vdd\_stop = 0.05
- l\_ax\_standard = '0.12u'
- l\_pmos\_q\_standard = '0.12u'
- w\_pmos\_q\_standard = '0.12u'
- l\_nmos\_q\_standard = '0.12u'
- w\_nmos\_q\_standard = '0.48u'
- l\_pmos\_q\_neg\_standard = '0.12u'
- w\_pmos\_q\_neg\_standard = '0.12u'
- l\_nmos\_q\_neg\_standard = '0.12u'
- w\_nmos\_q\_neg\_standard = '0.48u'
- vdd\_standard = '1'
- vsweep\_standard = '1'
- dc\_vsweep\_start\_standard = 0
- dc\_vsweep\_stop\_standard = 1
- dc\_vsweep\_step\_standard = 0.01
- v\_sweep\_seevinck = 0.655
- l\_ax\_seevinck = '0.12u'
- l\_pmos\_q\_seevinck = '0.12u'
- w\_pmos\_q\_seevinck = '0.12u'
- l\_nmos\_q\_seevinck = '0.12u'
- w\_nmos\_q\_seevinck = '0.48u'

- l\_pmos\_q\_neg\_seevinck = '0.12u'
- w\_pmos\_q\_neg\_seevinck = '0.12u'
- l\_nmos\_q\_neg\_seevinck = '0.12u'
- w\_nmos\_q\_neg\_seevinck = '0.48u'
- vdd\_seevinck = '1'
- vsweep\_seevinck = '1'
- e1\_seevinck = '0.707'
- e2\_seevinck = '0.707'
- e3\_seevinck = '1'
- e4\_seevinck = '1.41'
- e5\_seevinck = '-1'
- e6\_seevinck = '1.41'
- e7\_seevinck = '0.707'
- e8\_seevinck = '-0.707'
- l\_ax\_gaussian\_vth = '0.12u'
- w\_ax\_gaussian\_vth = '0.13u'
- l\_pmos\_q\_gaussian\_vth = '0.12u'
- w\_pmos\_q\_gaussian\_vth = '0.12u'
- l\_nmos\_q\_gaussian\_vth = '0.12u'
- w\_nmos\_q\_gaussian\_vth = '0.48u'
- l\_pmos\_q\_neg\_gaussian\_vth = '0.12u'
- w\_pmos\_q\_neg\_gaussian\_vth = '0.12u'
- l\_nmos\_q\_neg\_gaussian\_vth = '0.12u'
- w\_nmos\_q\_neg\_gaussian\_vth = '0.48u'
- vdd\_gaussian\_vth = '1'
- vsweep\_gaussian\_vth = '1'
- e1\_gaussian\_vth = '0.707'
- e2\_gaussian\_vth = '0.707'
- e3\_gaussian\_vth = '1'
- e4\_gaussian\_vth = '1.41'
- e5\_gaussian\_vth = '-1'
- e6\_gaussian\_vth = '1.41'
- e7\_gaussian\_vth = '0.707'
- e8\_gaussian\_vth = '-0.707'
- step\_param\_run\_gaussian\_vth\_start = 1
- step\_param\_run\_gaussian\_vth\_stop = 10000
- step\_param\_run\_gaussian\_vth\_step = 1
- l\_ax\_standard\_ileak = '0.12u'
- w\_ax\_standard\_ileak = '0.13u'
- l\_pmos\_q\_standard\_ileak = '0.12u'
- w\_pmos\_q\_standard\_ileak = '0.12u'
- l\_nmos\_q\_standard\_ileak = '0.12u'
- w\_nmos\_q\_standard\_ileak = '0.48u'
- l\_pmos\_q\_neg\_standard\_ileak = '0.12u'
- w\_pmos\_q\_neg\_standard\_ileak = '0.12u'
- l\_nmos\_q\_neg\_standard\_ileak = '0.12u'
- w\_nmos\_q\_neg\_standard\_ileak = '0.48u'

- vdd\_standard\_i leak = '1'
- vsweep\_standard\_i leak = '1'
- tran\_standard\_range = '150n'
- vbl\_hold\_i leak\_von = '0'
- vbl\_hold\_i leak\_tdelay = '100p'
- vbl\_hold\_i leak\_trise = '10p'
- vbl\_hold\_i leak\_tf fall = '10p'
- vbl\_hold\_i leak\_ton = '490p'
- vbl\_hold\_i leak\_tperiod = '1n'
- vblneg\_hold\_i leak\_von = '0'
- vblneg\_hold\_i leak\_tdelay = '100p'
- vblneg\_hold\_i leak\_trise = '10p'
- vblneg\_hold\_i leak\_tf fall = '10p'
- vblneg\_hold\_i leak\_ton = '490p'
- vblneg\_hold\_i leak\_tperiod = '1n'
- step\_param\_run\_standard\_i leak\_start = 1
- step\_param\_run\_standard\_i leak\_stop = 10000
- step\_param\_run\_standard\_i leak\_step = 1

## Global Functions



```

...
def snm_max(v_sweep):
    return f'.measure dc snm_max MAX({v_sweep}*v(V1)-{v_sweep}*v(V2))'

def snm_min(v_sweep):
    return f'.measure dc snm_min MIN({v_sweep}*v(V1)-{v_sweep}*v(V2))'
...
def dc_vsweep_gaussian_vth(dc_vsweep_start_gaussian_vth, dc_vsweep_stop_gaussian_vth, dc_vsweep_step_gaussian_vth):
    return f'.dc Vsweep {dc_vsweep_start_gaussian_vth} {dc_vsweep_stop_gaussian_vth} {dc_vsweep_step_gaussian_vth}'
...
def dc_vsweep_scaling(vdd_scaled):
    return f'.dc Vsweep {0} {vdd_scaled} {vdd_scaled/100}'
...

```

## I. Tasks to be performed

- a) Dimensionare opportunamente i dispositivi di una cella di memoria a 6T in maniera tale da avere dei ragionevoli margini di rumore in fase di hold e di read per un valore di tensione di alimentazione pari a 1V. Utilizzare LTSPICE e la metodologia grafica basata sull'analisi delle "curve a farfalla";
- b) Per il dimensionamento fissato al punto 1, ripetere l'analisi dei margini di rumore in fase di hold e read utilizzando il "metodo di Seevinck";
- c) Prevedere per i dispositivi un modello SPICE in cui la tensione di soglia assume una distribuzione di probabilità di tipo gaussiano, in maniera tale da mimare l'effetto delle variazioni di processo interdie. Utilizzando tale modello in LTSPICE, effettuare delle analisi MonteCarlo abbassando la tensione di alimentazione con degli step fissati. In particolare, per ogni step di tensione utilizzato, tabellare il valore medio e la varianza di:
  - potenza di leakage
  - margine di rumore in fase di hold
  - margine di rumore in fase di readInoltre, per ogni step di tensione di cui sopra, graficare la distribuzione di probabilità ottenuta per la corrente di leakage, il margine di rumore in fase di hold, il margine di rumore in fase di read. Fare le opportune considerazioni.
- d) Utilizzando un setup di simulazione opportuno, ricavare il grafico della distribuzione di probabilità del DRV

## II. Abstract

Nella parte iniziale del progetto è prevista la progettazione di una cella di memoria SRAM 6T tramite l'ambiente di sviluppo LTspice. Nello specifico, tale fase ha incluso il dimensionamento opportuno dei transistor associati e la corrispondente generazione delle curve utili al fine di calcolare il margine di rumore associato all'operazione considerata. Infatti, le analisi condotte hanno previsto sia una situazione di cella in standby, cioè in fase di hold, sia una situazione di lettura. In particolare, facendo riferimento una tensione di alimentazione pari a 1V, è stata condotta un'analisi mediante metodo grafico, cioè uno studio valutando l'andamento delle curve corrispondenti e il calcolo di SNM associato considerando il quadrato inscritto all'interno del minimo lobo tra le due curve.

Nella seconda parte del progetto è prevista la medesima analisi sopra citata ma considerando il metodo proposto nell'articolo ([Static-noise margin analysis of MOS SRAM cells, 1987](#)). Nello specifico, sfruttando la differenza tra le due curve ottenute tramite una nuova configurazione di circuito, è possibile calcolare direttamente tramite tale schematic il margine di rumore senza dover far riferimento al metodo grafico precedentemente citato.

Nella terza parte, invece, è prevista la  $V_{DD}$  scaling analysis. Nello specifico, tramite tale studio è stato possibile analizzare l'andamento della potenza di leakage, del margine di rumore di hold e di read per differenti valori di tensione di alimentazione. In particolare, per ogni step di  $V_{DD}$  è stato ottenuto il grafico corrispondente ad ogni parametro considerato così da effettuare le opportune considerazioni.

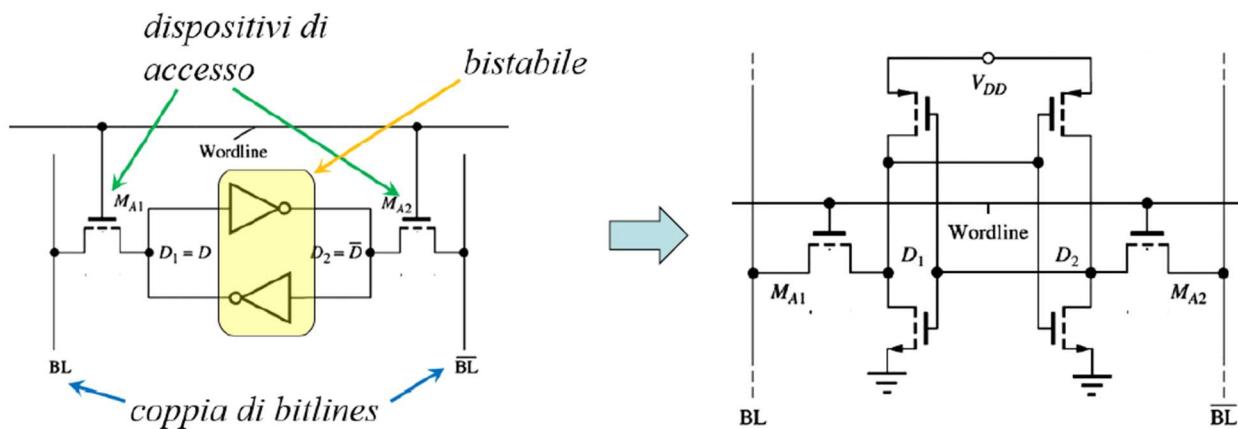
Nella parte finale del progetto è prevista l'analisi del parametro di Data Retention Voltage. Nello specifico, è stato ottenuto il grafico corrispondente al DRV e in seguito è stato possibile fare alcune considerazioni a riguardo così da scegliere un opportuno trade-off che rispettasse determinati vincoli di progetto.

# 1. Introduction

In questo capitolo verranno introdotti alcuni concetti teorici utili ad affrontare le tematiche e le analisi svolte all'interno del report del progetto.

## 1.1. SRAM 6T

Le celle di memoria di tipo SRAM rappresentano una componente cruciale nell'ambito dei circuiti integrati, svolgendo un ruolo fondamentale nella memorizzazione temporanea di informazioni all'interno di dispositivi elettronici complessi. Tra le diverse varianti di celle SRAM, la configurazione a 6 transistor (6T) in tecnologia CMOS (Complementary Metal-Oxide-Semiconductor) emerge come una delle soluzioni più ampiamente adottate grazie alla sua combinazione di prestazioni, efficienza energetica e scalabilità.



La cella SRAM è costituita da un bistabile e da due dispositivi di accesso, cioè i transistor di accesso indicati nella figura sopra allegata con le denominazioni  $M_{A1}$  e  $M_{A2}$ , tramite i quali è possibile effettuare la lettura e la scrittura. Inoltre, è presente una coppia di bitline come per ogni colonna della memoria complessiva. Nello specifico, il bistabile, cioè l'elemento base dei circuiti di memoria statici, è costituito da due invertori CMOS.

All'interno di questo progetto verranno analizzate la fase di mantenimento del dato, cioè la Hold Phase, e la fase di lettura del dato, cioè la Read Phase.

## 1.2. Hold Phase

La fase di Hold della cella SRAM prevede che la word line sia connessa a GND. Nello specifico, la coppia di inverter (*bistable cross-coupled inverters*) deve mantenere il punto di funzionamento bistabile così da preservare correttamente il proprio stato e, quindi, conservare il dato. In particolare, essendo  $WL=0$  si ha che i transistor di accesso risultano essere spenti così da garantire che la memoria non sia accessibile né per operazioni di lettura né per operazioni di scrittura.

## 1.3. Read Phase

La fase di Read della cella SRAM prevede che la word line sia impostata sul valore di  $V_{DD}$ . In questo modo i transistor di accesso risulteranno essere accesi così da garantire un accesso di memoria per tale operazione.

Inizialmente, attraverso un circuito ausiliario, le capacità parassite delle due bitline vengono precaricate a una tensione prefissata mentre la wordline è mantenuta a zero per non disturbare la cella di memoria. Una volta completata la precarica, la wordline viene attivata, causando lo scaricamento della capacità sulla bitline

sinistra a 0V e la carica di quella sulla bitline destra a  $V_{DD}$ . Successivamente, i potenziali sulle bitline si bilanciano, e la differenza di potenziale associata viene amplificata da un amplificatore di lettura (sense amplifier) che fornisce l'uscita.

## 1.4. SNM

Lo Static Noise Margin SNM è un parametro caratteristico delle memorie SRAM che permette di valutare il grado di robustezza del dispositivo stesso. In particolare, esistono tre differenti margini di rumore: SNM(HOLD), SNM(READ) e SNM(WRITE). Nello specifico, rappresentano rispettivamente la facilità di mantenere il dato all'interno della cella, la facilità di leggere un dato all'interno della cella e la facilità di scrivere all'interno della cella.

La facilità o la difficoltà di mantenimento del dato nella cella è dovuto allo studio di una possibile tecnica che possa mantenere correttamente il dato all'interno di essa. Questo è dovuto a possibili problemi riguardanti eventuali variazioni di processo, interferenze elettromagnetiche o, ad esempio, ipotetici picchi di tensione. Pertanto, più è robusta la cella in fase di Hold più è difficile che eventi come quelli appena citati possano perturbare il comportamento o il contenuto della stessa.

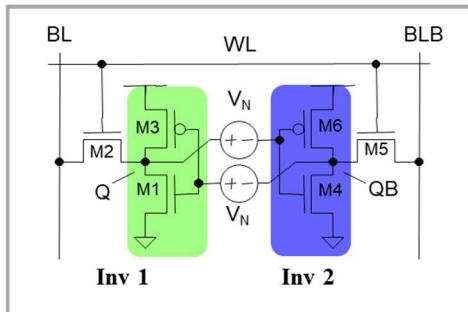
La facilità o la difficoltà della lettura per una cella di questo genere potrebbe essere dovuta alla gestione delle bitline corrispondenti. Bisogna ricordare che tali linee di tensione, in fase di Read, sono mantenute entrambe a  $V_{DD}$  tale che durante l'operazione in questione una delle due possa scaricarsi. In particolare, se una delle due bitline si scarica non correttamente, questo potrebbe comportare picchi di tensione e tali picchi potrebbero essere ingigantiti dalla retroazione positiva presente all'interno della cella. Nello specifico, un'eventualità del genere potrebbe comportare una lettura non corretta del dato e in casi peggiori potrebbe provocare il flipping del dato stesso.

La facilità o la difficoltà della scrittura per una cella SRAM è dovuta alla presenza del feedback positivo presente all'interno della cella. Tale retroazione permette alla stessa un corretto mantenimento del dato durante la fase di sleep. In particolare, la difficoltà è determinata dalla possibilità di vincere la "forza oppositrice" della retroazione appena citata.

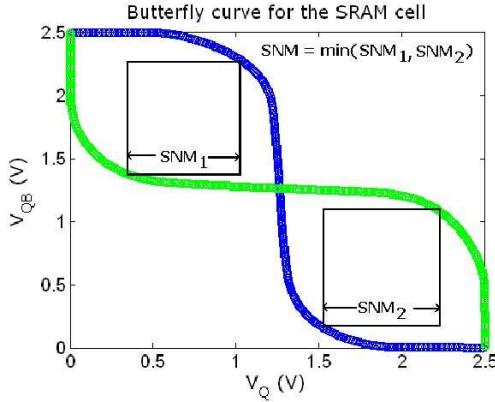
Per quanto riguarda il calcolo effettivo del margine di rumore, si fa riferimento all'analisi dei lobi ottenuti dalla cella corrispondente. In particolare, si prendono in considerazione le seguenti due tecniche: il Graphical Method e il Seevinck Method.

### 1.4.1. Graphical Method

Il metodo grafico prevede il calcolo dell'SNM mediante analisi grafica dei lobi corrispondenti ai nodi Q e Qneg. In particolare, la tecnica consiste nel posizionare in corrispondenza del nodo Q un generatore  $V_N$  e successivamente di plottare nello stesso grafico la curva ottenuta e la sua simmetrica.



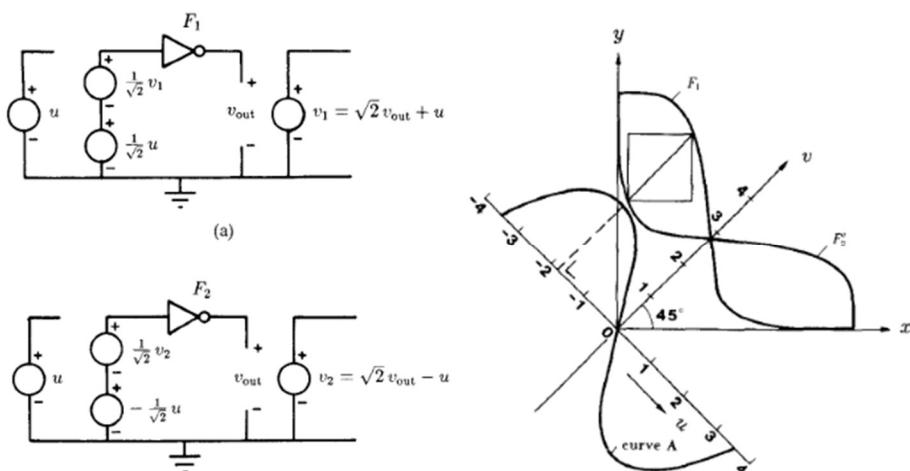
In questo modo, il grafico risultante sarà composto da due lobi dai quali è possibile effettuare graficamente delle analisi per giungere al calcolo effettivo del parametro di SNM. In particolare, il valore di margine di rumore sarà dato dalla lunghezza del quadrato inscritto all'interno del lobo più piccolo. Ovviamente, per ottenere un corretto calcolo di tale parametro, l'asse delle ascisse e quello delle ordinate dovranno essere dimensionati allo stesso modo.



Questa tipologia di analisi, dal momento che si tratta di uno studio grafico, risulta essere piuttosto imprecisa. Nel caso in cui, comunque, si voglia considerare un parametro di riferimento approssimativo riguardo il margine di rumore della cella, tale metodo risulta essere quello più veloce e semplice tra quelli possibili.

#### 1.4.2. Seevinck Method

Il metodo di Seevinck permette di calcolare il margine di rumore considerando direttamente una configurazione circuitale che calcola in "automatico" tale parametro. Nello specifico, non si dovrà considerare una successiva analisi basata sul metodo grafico ma basterà calcolare il massimo o il minimo della differenza tra le curve ottenute in corrispondenza dei nodi Q e Qneg. In particolare, la differenza tra le due curve corrisponde ad un segnale di tensione. Il valore assoluto del massimo e del minimo di tale segnale corrispondono rispettivamente alla diagonale del quadrato inscritto nel lobo più grande e di quello inscritto nel lobo più piccolo.



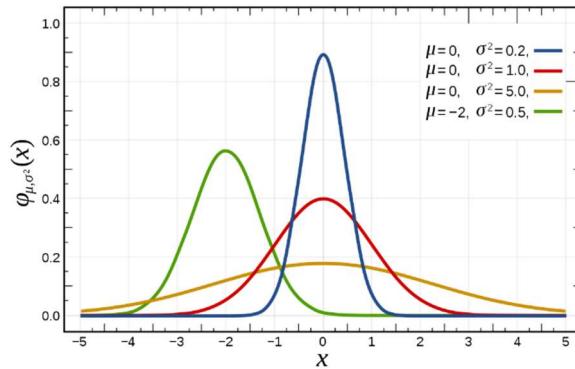
Pertanto, il valore di SNM sarà pari al prodotto tra il minimo tra i due parametri appena citati e la costante  $\frac{1}{\sqrt{2}}$ . Questo prodotto è dovuto al fatto che il valore di SNM corrisponde al lato del quadrato più piccolo tra i due inscrivibili rispettivamente nei due lobi. Pertanto, a partire dal valore della diagonale, ottenuto tramite la configurazione circuitale sopra citata, è possibile ottenere il valore del lato del quadrato e, pertanto, il valore di SNM corrispondente alla cella in questione.

## 1.5. Normal Distribution

La distribuzione normale, anche conosciuta come distribuzione gaussiana, è un tipo di distribuzione di probabilità continua basata su una variabile random a valore reale. La densità di probabilità corrispondente presenta la seguente funzione:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

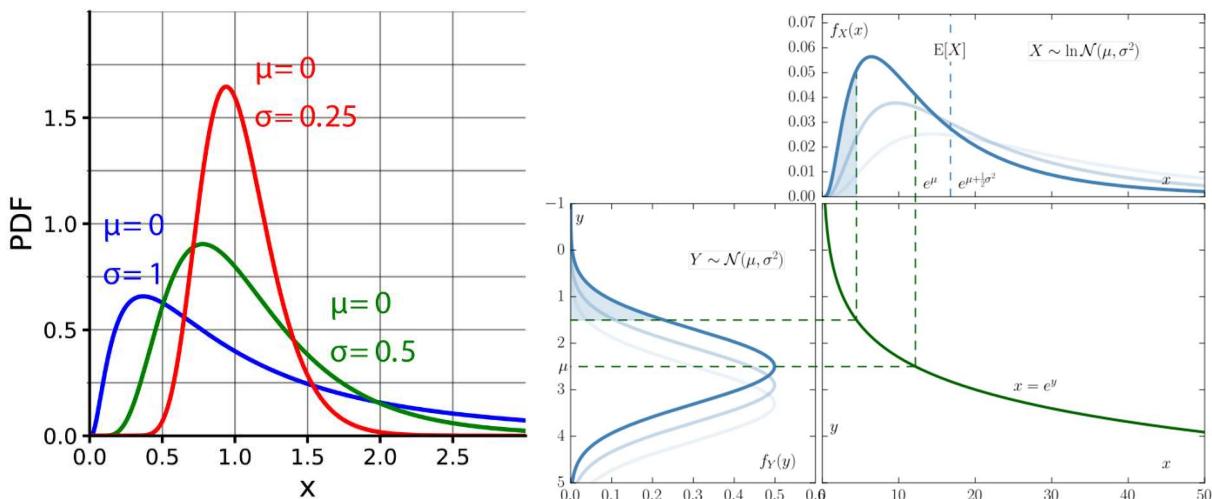
Il parametro  $\mu$  è la media della distribuzione mentre il parametro  $\sigma$  è la deviazione standard. La varianza della distribuzione è rappresentata dal parametro  $\sigma^2$ .



L'importanza della distribuzione normale è dovuta al teorema del limite centrale. Esso afferma che, in alcune condizioni, la media di un insieme di campioni (osservazioni) di una variabile casuale, avente media e varianza finite, è essa stessa una variabile casuale la cui distribuzione, all'aumentare dei campioni considerati, converge ad una distribuzione normale.

## 1.6. Log-Normal Distribution

La distribuzione log-normale è una distribuzione di probabilità continua di una variabile casuale il cui logaritmo è distribuito in modo normale. Pertanto, se la variabile casuale  $X$  è distribuita in modo log-normale, allora  $Y = \ln(X)$  ha una distribuzione normale. Equivalentemente, se  $Y$  ha una distribuzione normale, allora la funzione esponenziale di  $Y$ ,  $X = e^Y$ , ha una distribuzione log-normale. Pertanto, una variabile casuale con distribuzione log-normale assume solo valori reali positivi.



Sia  $Z$  una variabile normale e siano  $\mu$  e  $\sigma$  due numeri reali tale che  $\sigma > 0$ , allora la distribuzione della variabile casuale

$$X = e^{\mu + \sigma Z}$$

è chiamata distribuzione log-normale tale che i parametri  $\mu$  e  $\sigma$  sono rispettivamente la media e la deviazione standard del logaritmo naturale della variabile e non la media e la deviazione standard della variabile stessa.

## 1.7. $V_{DD}$ Scaling

Lo scaling della  $V_{DD}$  viene effettuato per ridurre la potenza di leakage associata alla cella. Nello specifico, quando la cella risulta essere in fase di standby, la tensione di alimentazione viene impostata su una  $V_{SLEEP}$ , mentre quando viene acceduta allora viene impostata su  $V_{ACTIVE}$ . Pertanto, il  $V_{DD}$  scaling si tratta di una tecnica di optimizing power di standby tale che, quando la cella si trova in fase di active viene disabilitata mentre quando si trova in fase di standby viene abilitata.

Il vantaggio di tale tecnica è ridurre il leakage associato alla cella, cioè la riduzione delle tre correnti di leakage. Questo aspetto è molto importante poiché il leakage associato alla singola cella deve essere considerato in un'ottica tale che questo sarà associato a quasi la totalità dell'array di celle corrispondente alla memoria. Nella maggior parte dei casi, l'insieme delle celle che vengono accedute in un determinato istante temporale risulta essere ridotto rispetto alla totalità delle celle di cui è composta una memoria. Pertanto, in un determinato intervallo temporale, soltanto una piccola porzione di questo insieme risulta essere in active mentre la maggior parte delle celle della memoria risulterà essere in fase di sleep comportando, quindi, un certo quantitativo di leakage. Tale potenza di leakage, pertanto, se non ridotta, considerando l'insieme delle celle, appena citate, risulterà essere enorme. Di conseguenza, applicare una tecnica o più tecniche di riduzione del leakage nella cella è di fondamentale importanza.

## 1.8. DRV

Lo scaling della  $V_{DD}$  prevede, quindi, la diminuzione della tensione di alimentazione. In particolare, tale riduzione avviene per step, cioè si sceglie un passo di scaling e gradualmente, scalando tale tensione, si verifica il corretto funzionamento della cella. Nello specifico, il corretto funzionamento della cella può essere analizzato in diversi modi. Ad esempio, si possono verificare i valori di SNM associati alla singola cella oppure verificare il corretto funzionamento di un gruppo di celle che fanno da "cavia" (denominate *canary cells*) per il resto delle celle. Pertanto, procedendo con uno scaling continuo, nel momento in cui vengono individuati malfunzionamenti in tali celle o nel momento in cui il margine di rumore risulta essere inferiore ad una certa soglia (solitamente pari a  $60\text{ mV}$ ), la tensione  $V_{DD}$  viene riportata ad un valore che permette il corretto funzionamento delle celle. In particolare, tale valore limite, oltre il quale le celle iniziano a fallire, è denominato DRV (Data Retention Voltage). Pertanto, lo scaling della tensione di alimentazione non dovrebbe oltrepassare il valore del parametro appena citato per evitare malfunzionamenti alle celle associate. In particolare, il valore di  $V_{DD}$  deve essere scalato rispettando la seguente relazione:

$$V_{DD} > DRV$$

Nello specifico, il valore della tensione di alimentazione, teoricamente, non deve essere nemmeno uguale al valore limite dal momento che, in caso di temperatura più alta o altre tipologie di variazioni, il valore di DRV potrebbe cambiare e, pertanto, provare fallimenti di celle improvvisi. Infatti, solitamente il valore di tensione associato alla  $V_{SLEEP}$  viene scelto nel seguente modo:

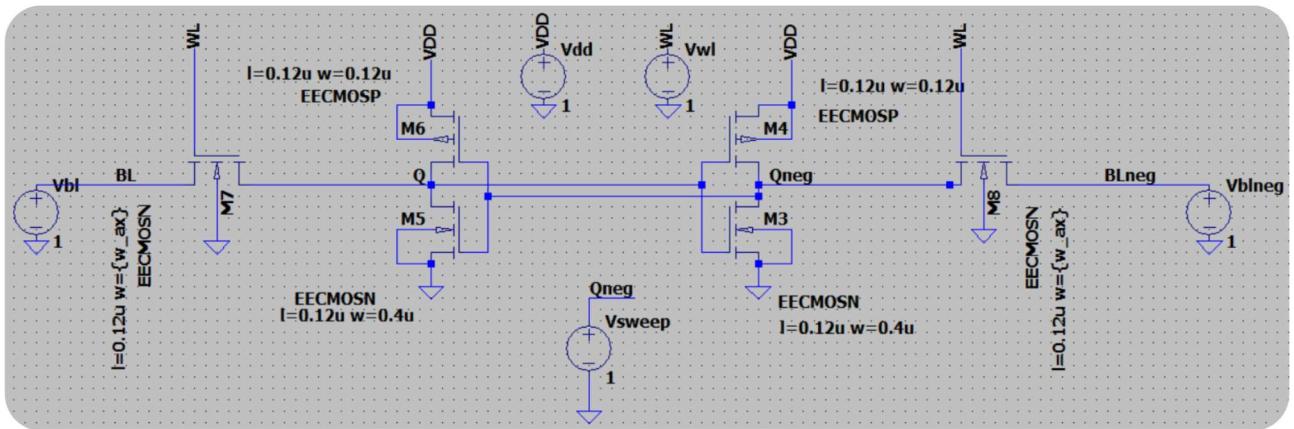
$$V_{SLEEP} = DRV + security\_margin$$

Soltamente come valore di margine di sicurezza si sceglie  $100\text{ mV}$ , peggiorando di conseguenza il leakage. Però così facendo viene permesso, teoricamente, alla memoria di non fallire a causa dello scaling della  $V_{DD}$ .

Nel caso in cui si voglia scegliere un valore di  $V_{DD}$  uguale o addirittura inferiore al valore limite di DRV, si può optare per soluzioni di memorie basate su tecniche di ECC, cioè Error Correction Code. In particolare, tali strategie permettono di far fallire le celle e allo stesso tempo, nel caso in cui i dati vengano corrotti, di identificare l'errore, quindi fare detection, e correggerlo. Ovviamente, per poter applicare tale tecnica, bisogna prevedere l'aggiunta di bit di parità alla parola, compito che viene svolto da un componente circuitale denominato encoder. Per quanto riguarda la correction del dato, essa viene effettuata mediante un ulteriore componente, denominato decoder, che permette, tramite gli stessi bit di parità, di isolare l'errore e correggerlo così che la parola corretta possa essere riportata in memoria e, inoltre, prevedendo dei nuovi bit di parità che potrebbero servire in futuro nuovamente per identificare un eventuale nuovo errore.

## 2. Analysis and Design of a 6T SRAM via Graphical Method

L'analisi mediante metodo grafico consiste nel plottare le curve relative ai segnali Q e Qneg e successivamente inscrivere all'interno dei lobi, creati in seguito all'introduzione di tali curve all'interno del plot, un quadrato. Tale quadrato deve essere il più grande che si possa inscrivere all'interno del lobo più piccolo tra i due possibili. Il lato del quadrato inscritto corrisponderà al margine di rumore statico SNM. Tale metodo risulta essere impreciso ma permette comunque un'analisi veloce del SNM senza disporre di circuiti di analisi più avanzati come, ad esempio, il circuito di Seevinck. Ovviamente, affinché la misura venga effettuata correttamente, l'asse delle ascisse e quello delle ordinate devono essere dimensionati allo stesso modo e si deve disporre di uno strumento di misura preciso ed affidabile per la misura del lato del quadrato come quello disponibile in Excel o in altri software analoghi. In questo progetto, l'analisi mediante metodo grafico è stata implementata in Python e nello specifico è stato sviluppato un algoritmo che, date le curve Q e Qneg, restituisce in automatico il valore di SNM senza fare uso di software esterni come Excel.



Dallo schematic in questione, si può notare come il generatore Vsweep è stato collegato al nodo Qneg così da ottenere, tramite simulazione, la curva  $V(Q)$  per le analisi riguardo il SNM. Ovviamente un ragionamento analogo si potrebbe fare considerando un generatore Vsweep con le medesime caratteristiche ma collegato al nodo Q. In questo modo, invece, si ottiene, tramite simulazione, la curva  $V(Q_{neg})$ .

### 2.1. Hold Phase

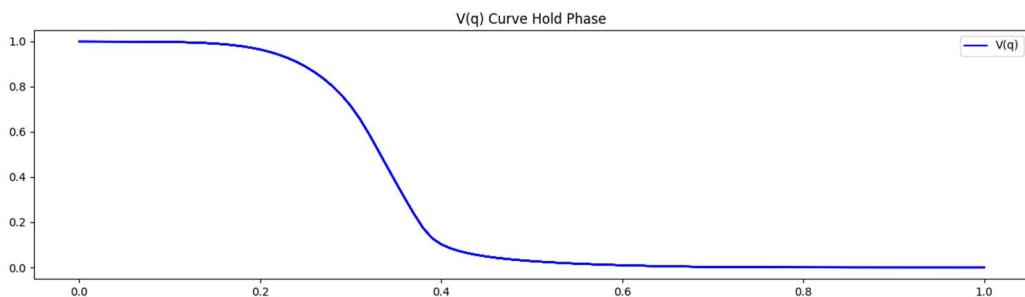
Per l'analisi di Hold è stato utilizzato lo script `snm_hold_standard_analysis.py` nel quale è presente il metodo `__init_model__` e dove sono stati specificati il valore STANDARD come `CircuitType` e il valore HOLD come `OperationType`. Inoltre, sono state specificate le costanti `vdd_standard`, `vsweep_standard`, `vwl_hold`, `vbl_hold` e `vblneg_hold` rispettivamente per i parametri in input `vdd`, `vsweep`, `vwl`, `vbl` e `vblneg`.

```

    (
        steps,
        vsweep_standard_hold,
        i_leaks_standard_hold,
        v_q_standard_hold,
        v_q_neg_standard_hold,
        v_dd_standard_hold,
        v_wl_standard_hold,
        v_b1_standard_hold,
        v_blneg_standard_hold,
        standard_hold_log
    ) = __init_model__(
        operation_type=OperationType.HOLD,
        circuit_type=CircuitType.STANDARD,
        asc_file_path=os.path.join(ltspice, "standard/hold/standard_hold.asc"),
        schematic_image_path=os.path.join(schematics, "standard.png"),
        request_plot_schematic=RequestPlotSchematic.TRUE,
        vdd=vdd_standard,
        vsweep=vsweep_standard,
        vwl=vwl_hold,
        vbl=vbl_hold,
        vblneg=vblneg_hold,
        params=[rit_models, dc_vsweep_standard, w_ax_step_param_standard, save_w_ax_standard]
    )
)

```

Simulando lo schematico sopra allegato secondo i parametri appena citati otterremo la seguente curva:

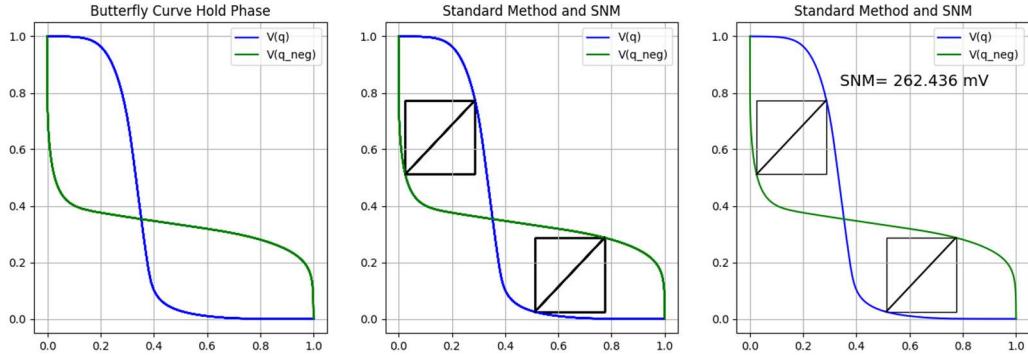


Nello specifico, le curve ottenute corrispondono alla  $V(Q)$ , ognuna corrispondente alla  $w_{AX}$  relativa. Ognuna di queste curve verrà posta in input all'algoritmo progettato per il calcolo grafico dell'SNM. Pertanto, invocando il metodo graphical\_processing nel range di larghezza del canale relativa al transistor di accesso, si calcolerà il valore di SNM per ogni  $W_{AX}$  considerata.

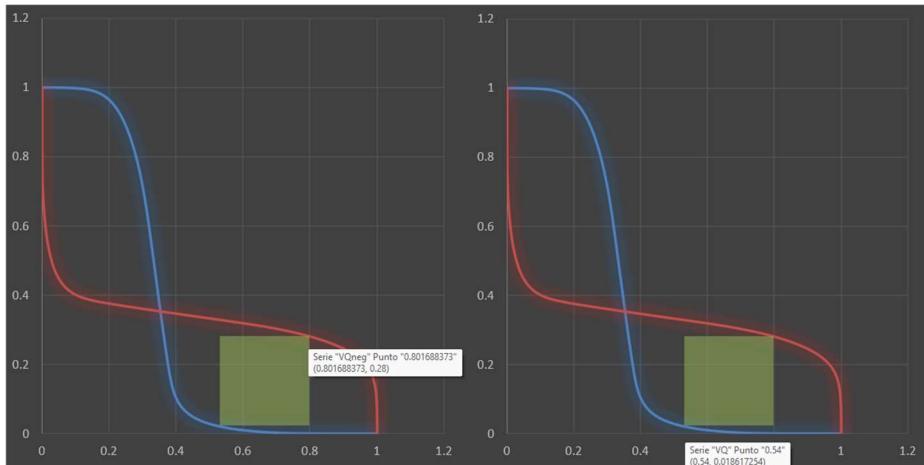
```

snm_standard_hold = []
for w in range(w_ax_range):
    snm_standard_hold_value = graphical_processing(
        x_vq=x_vq_standard_hold[w],
        vq=vq_standard_hold[w],
        x_vqneg=x_vqneg_standard_hold[w],
        vqneg=vqneg_standard_hold[w],
        ax=axs[1],
        request_text_plot=RequestPlot.FALSE
    )
    snm_standard_hold.append(snm_standard_hold_value)

```

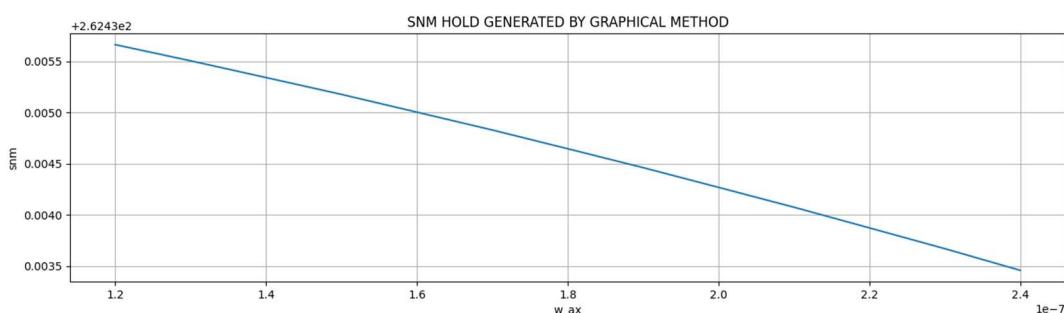


Nel terzo subplot sono state mostrate, per semplicità, soltanto le curve ed il valore di SNM relative alla larghezza di canale del transistor di accesso pari a  $0.13\mu m$ , cioè per  $w_{ax\_pos} = 1$  come specificato nelle definizioni iniziali. Per confermare i calcoli effettuati dall'algoritmo progettato in Python si mostrano qui di seguito le curve corrispondenti ed il calcolo di SNM effettuato in Excel:



Si può notare come effettivamente alle ascisse corrispondenti agli estremi del lato del quadrato corrisponde una lunghezza del lato e, pertanto, un SNM pari a  $0.801688373 - 0.54 = 0.261688373$ . Tale valore, infatti, lo si può considerare praticamente uguale al precedente ottenuto tramite Python dal momento che era pari a 0.262436.

Nel secondo subplot generato tramite Python, invece, sono mostrate le curve relative a tutti i valori di  $w_{ax}$  incluse nel range specificato nelle definizioni. Si può notare che tali curve e, di conseguenza, anche i quadrati inscritti nei loro lobi risultano essere praticamente uguali per ogni  $w_{ax}$  considerata. Questo perché nella fase di hold, essendo che la word line risulta essere spenta, pur variando le caratteristiche del transistor di accesso dato che quest'ultimo risulta essere spento, le curve relative al nodo  $V(Q)$  o  $V(Qneg)$  risultano essere uguali o comunque leggermente differenti alle altre pur considerando una larghezza di canale diversa per il transistor di accesso. Tale aspetto lo si può analizzare meglio nel dettaglio nel seguente plot:



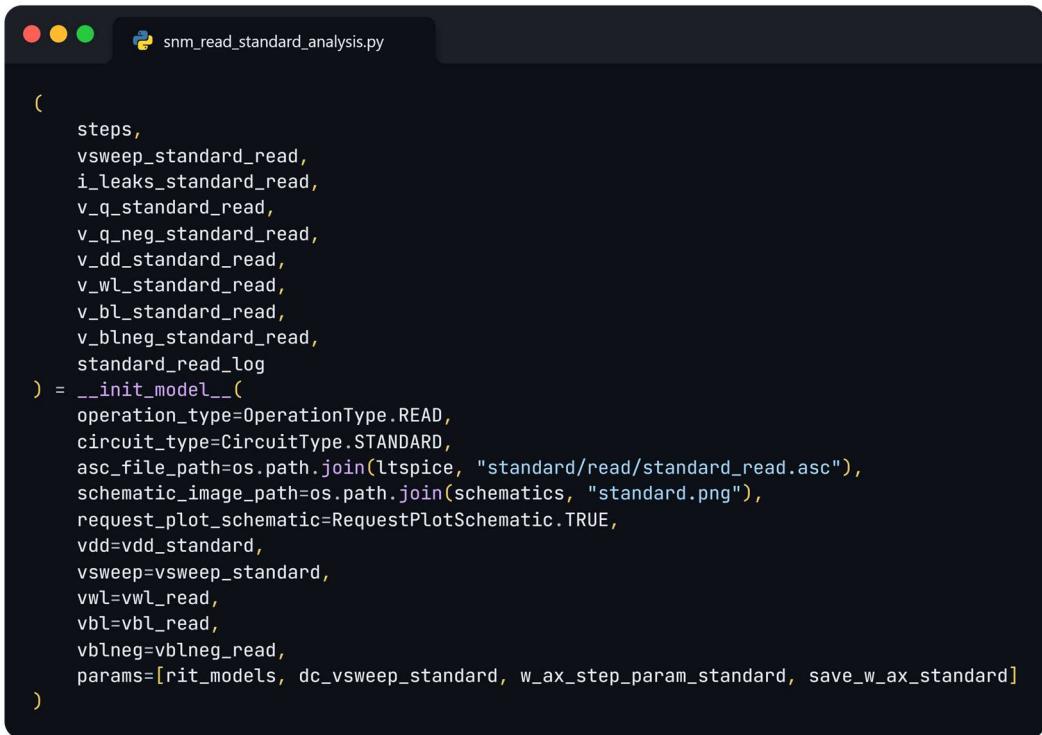
Si può notare, infatti, che, pur variando la  $w_{ax}$ , la variazione di SNM risulta essere praticamente quasi pari a zero tranne che per piccole variazioni nell'ordine di  $10^{-3}$ . La seguente tabella permette di analizzare più nel dettaglio quanto detto:

### Comparative Analysis W\_AX-SNM(HOLD) Graphical Method

w_ax [m]	snm(hold)_standard [mV]
1.2e-07	262.43566244928326
1.3e-07	262.43550473079256
1.4e-07	262.43534129914343
1.5e-07	262.43517705584117
1.6e-07	262.4350036798156
1.7e-07	262.43482954124966
1.8e-07	262.43464673132866
1.9e-07	262.4344622101111
2e-07	262.43426967969657
2.1e-07	262.4340759824953
2.2e-07	262.43387439480875
2.3e-07	262.43367158989145
2.4e-07	262.43346020573165

## 2.2. Read Phase

Per l'analisi di Read è stato utilizzato lo script `snm_read_standard_analysis.py` nel quale è presente il metodo `__init_model__` e dove sono stati specificati il valore STANDARD come CircuitType e il valore READ come OperationType. Inoltre, sono state specificate le costanti vdd\_standard, vsweep\_standard, vwl\_read, vbl\_read e vblneg\_read rispettivamente per i parametri in input vdd, vsweep, vwl, vbl e vblneg.

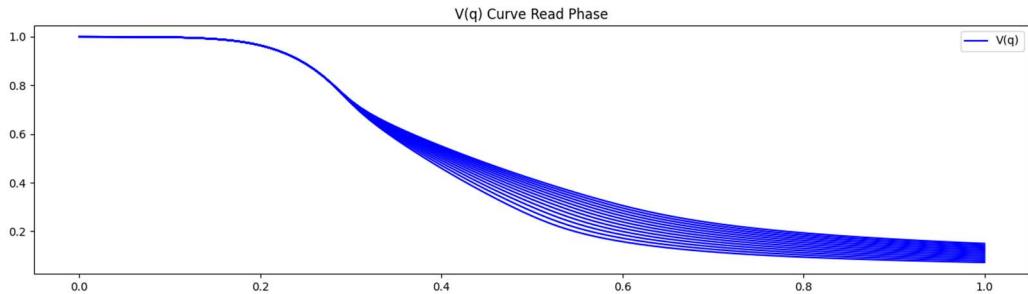


```

    (
        steps,
        vsweep_standard_read,
        i_leaks_standard_read,
        v_q_standard_read,
        v_q_neg_standard_read,
        v_dd_standard_read,
        v_wl_standard_read,
        v_bl_standard_read,
        v_blneg_standard_read,
        standard_read_log
    ) = __init_model__(
        operation_type=OperationType.READ,
        circuit_type=CircuitType.STANDARD,
        asc_file_path=os.path.join(ltspice, "standard/read/standard_read.asc"),
        schematic_image_path=os.path.join(schematics, "standard.png"),
        request_plot_schematic=RequestPlotSchematic.TRUE,
        vdd=vdd_standard,
        vsweep=vsweep_standard,
        vwl=vwl_read,
        vbl=vbl_read,
        vblneg=vblneg_read,
        params=[rit_models, dc_vsweep_standard, w_ax_step_param_standard, save_w_ax_standard]
    )

```

Simulando lo schematico precedentemente allegato secondo i parametri appena citati otterremo la seguente curva:

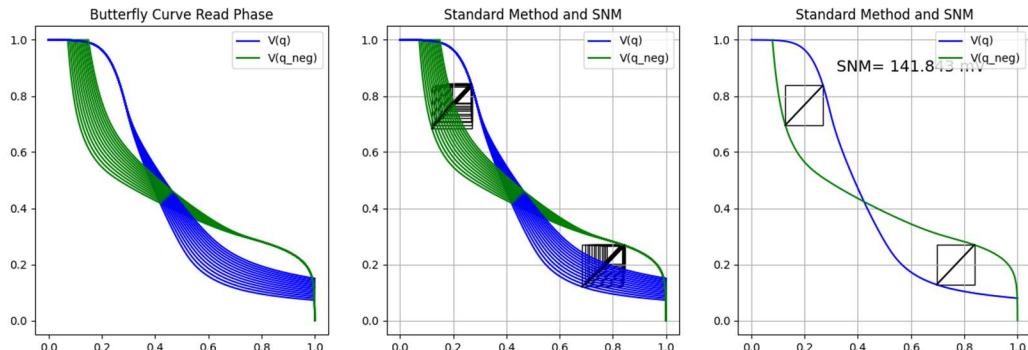


Rispetto alle curve generate durante la fase di hold, in questo caso le curve risultano essere differenti tra loro poichè essendo abilitata la word line, i transistor di accesso influenzano i nodi Q e Qneg. Pertanto, ponendo ognuna di queste curve, corrispondenti ognuna ad una  $w_{ax}$  differente, in input al metodo graphical\_processing sarà possibile calcolare il valore di SNM per ognuna considerando i quadrati inscrivibili all'interno dei lobi come precedentemente spiegato.

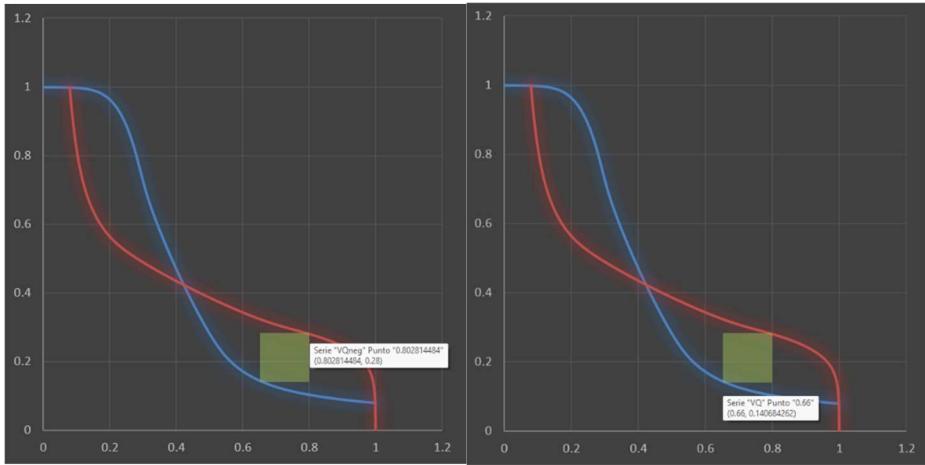
```

snm_standard_read = []
for w in range(w_ax_range):
    snm_standard_read_value = graphical_processing(
        x_vq=x_vq_standard_read[w],
        vq=vq_standard_read[w],
        x_vqneg=x_vqneg_standard_read[w],
        vqneg=vqneg_standard_read[w],
        ax= axs[1],
        request_text_plot=RequestPlot.FALSE
    )
    snm_standard_read.append(snm_standard_read_value)

```

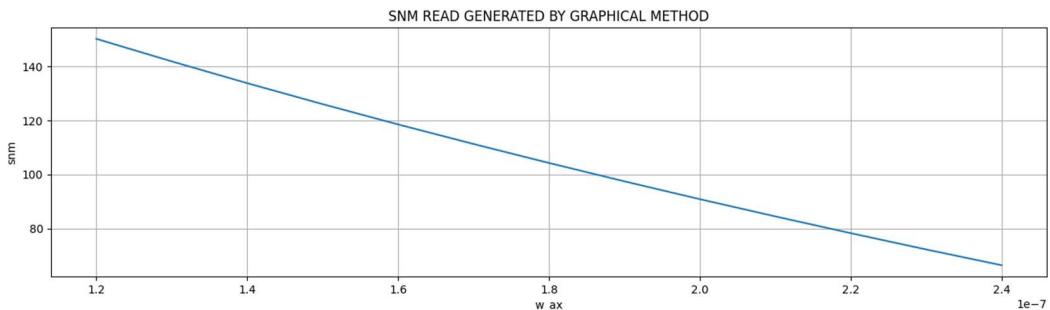


Per confermare i calcoli effettuati dall'algoritmo progettato in Python si mostrano qui di seguito le curve corrispondenti ed il calcolo di SNM effettuato in Excel:



Si può notare come effettivamente alle ascisse corrispondenti agli estremi del lato del quadrato corrisponde una lunghezza del lato e, pertanto, un SNM pari a  $0.802814484 - 0.66 = 0.142814484$ . Tale valore, infatti, lo si può considerare praticamente uguale al precedente ottenuto tramite Python dal momento che era pari a 0.141843.

Si può notare, infatti, come nel secondo subplot la grandezza dei quadrati e, quindi, i corrispondenti valori di SNM siano differenti per ogni curva considerata, cioè per ogni larghezza del canale del transistor di accesso considerata. Pertanto, graficando il valore di SNM al variare della  $w_{ax}$  ci si aspetta un cambiamento di margine di rumore importante.



In effetti, dal plot sopra allegato si può notare come nel range di larghezza del canale del transistor di accesso considerato, cioè  $[0.12, 0.24]\mu m$ , il valore di SNM cambia notevolmente. Rispettivamente da un valore di circa 150 mV si arriva ad un valore di circa 66 mV in corrispondenza di  $w_{ax} = 0.24 \mu m$ . Questo aspetto era già riscontrabile nel secondo subplot sopra allegato dove era possibile notare l'avvicinamento delle curve associate ai nodi Q e Qneg comportando, di conseguenza, una riduzione della diagonale del quadrato inscrivibile all'interno del lobo minore. La seguente tabella permette di analizzare più nel dettaglio i valori appena citati:

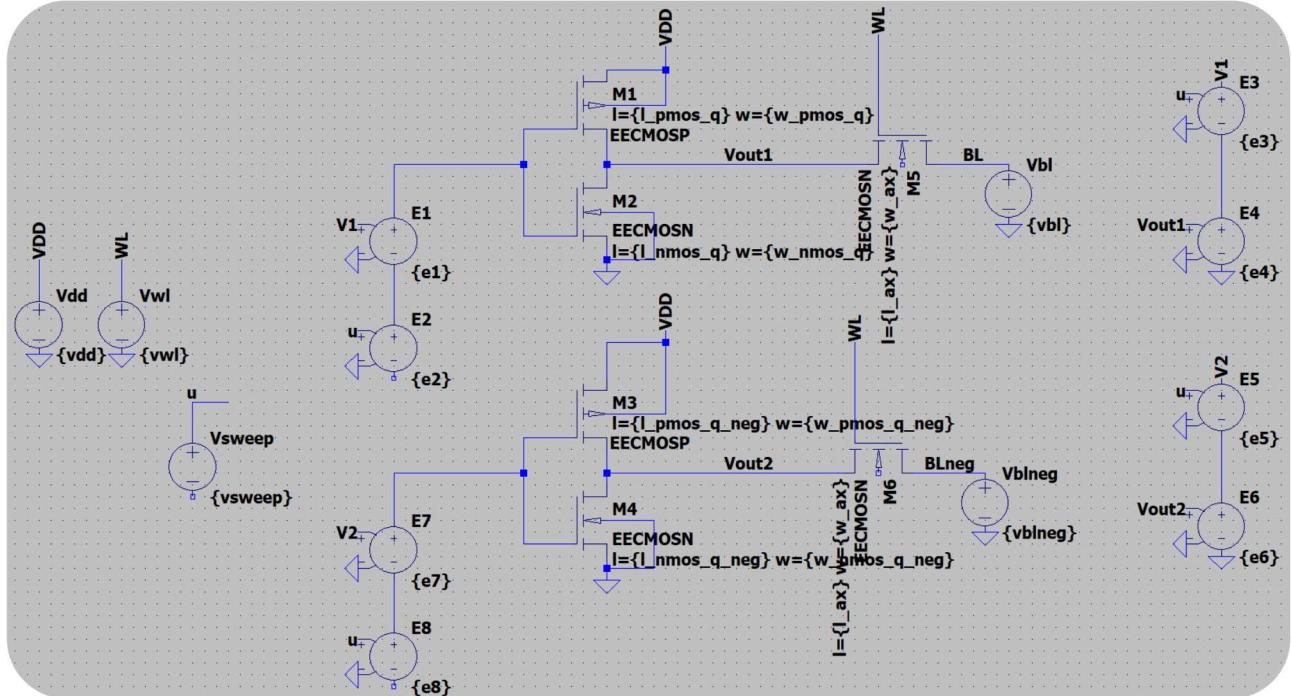
Comparative Analysis W\_AX-SNM(READ) Graphical Method

w_ax [m]	snm(read)_standard [mV]
1.2e-07	150.18932179736015
1.3e-07	141.84301463106343
1.4e-07	133.79837532531718
1.5e-07	126.04461719878147
1.6e-07	118.55191413438021
1.7e-07	111.31813478480773
1.8e-07	104.30266609752346
1.9e-07	97.51147039013014
2e-07	90.92296793564506
2.1e-07	84.55820794965902
2.2e-07	78.35772229810932
2.3e-07	72.33534210594759
2.4e-07	66.5079460850747

### 3. Analysis and Design of a 6T SRAM via Seevinck Method

Considerando il metodo di Seevinck che permette di calcolare il valore di SNM direttamente tramite una determinata configurazione di circuito, si effettuano gli stessi calcoli precedentemente esposti nel capitolo precedente e, inoltre, si effettua una comparazione tra il metodo grafico e il suddetto metodo di Seevinck.

Qui di seguito viene allegato la configurazione di circuito utilizzata per l'implementazione del metodo di Seevinck.



Dallo schematic sopra allegato si può notare come i generatori  $v_1$  e  $v_2$ , presenti nell'articolo di Seevinck, corrispondano rispettivamente al nodo  $v_1$  e  $v_2$ . Tali generatori, dal momento che dovrebbero rispettivamente implementare il calcolo matematico  $\sqrt{2} \cdot v_{out} + u$  e  $\sqrt{2} \cdot v_{out} - u$ , sono stati realizzati tramite generatori di tensione controllati in tensione.

L'obiettivo è quello di ottenere la differenza tra la curva  $v_1$  e la curva  $v_2$ , cioè  $v_1 - v_2$ , così da calcolare l'altezza di tale curva, poiché rappresenterà la diagonale del quadrato inscrivibile nel lobo più piccolo (facendo riferimento al metodo grafico precedentemente analizzato), e successivamente essere in grado di calcolare il valore di SNM con un banalissimo calcolo matematico. In questo caso è stato implementato in Python che permette di calcolare automaticamente il valore dell'altezza e il corrispondente valore di SNM. Ovviamente tali calcoli è possibile svolgerli in maniera accurata tramite una direttiva .meas all'interno di LTspice. Pertanto, nelle successive sezioni in cui si affronterà sia l'analisi della fase di mantenimento del dato sia la fase di lettura, verrà inizialmente presentato il calcolo mediante funzione .meas e successivamente confrontato con il valore ottenuto via software tramite algoritmo implementato in Python.

#### 3.1. Hold Phase

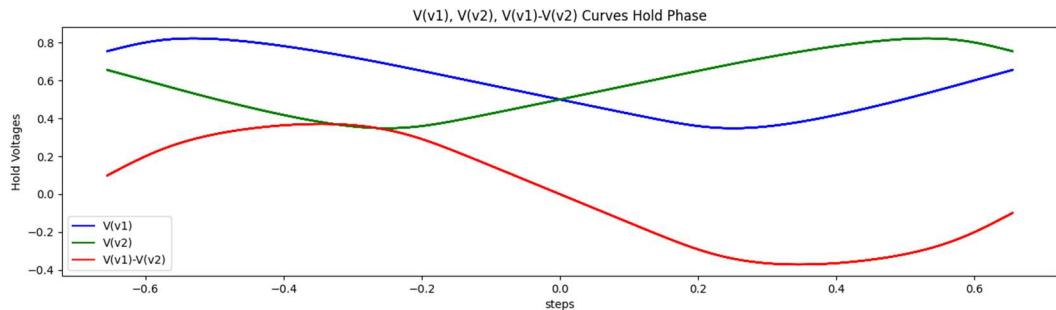
Per l'analisi di Hold è stato utilizzato lo script `snm_hold_seevinck_analysis.py` nel quale è presente il metodo `__init_model__` e dove sono stati specificati il valore SEEVINCK come `CircuitType` e il valore HOLD come `OperationType`. Inoltre, sono state specificate le costanti `vdd_seevinck`, `vsweep_seevinck`, `vwl_hold`, `vbl_hold` e `vblneg_hold` rispettivamente per i parametri in input `vdd`, `vsweep`, `vwl`, `vbl` e `vblneg`.

```

    (
        steps,
        vsweep_seevinck_hold,
        i_leaks_seevinck_hold,
        v_1_seevinck_hold,
        v_2_seevinck_hold,
        v_dd_seevinck_hold,
        v_wl_seevinck_hold,
        v_b1_seevinck_hold,
        v_blneg_seevinck_hold,
        seevinck_hold_log
    ) = __init_model__(
        operation_type=OperationType.HOLD,
        circuit_type=CircuitType.SEEVINCK,
        asc_file_path=os.path.join(LTspice, "seevinck/hold/seevinck_hold.asc"),
        schematic_image_path=os.path.join(schematics, "seevinck.png"),
        request_plot_schematic=RequestPlotSchematic.TRUE,
        vdd=vdd_seevinck,
        vsweep=vsweep_seevinck,
        vwl=vwl_hold,
        vbl=vbl_hold,
        vblneg=vblneg_hold,
        params=[rit_models, dc_vsweep_seevinck, snm_max(0.707), snm_min(0.707), w_ax_step_param_seevinck, save_w_ax_seevinck]
)

```

Simulando lo schematico sopra allegato secondo i parametri appena citati otterremo la seguente curva:

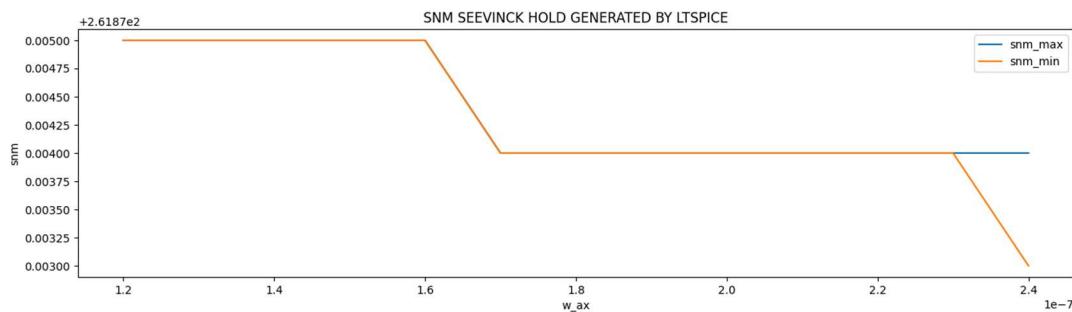


Nello specifico la curva blu generata corrisponde al segnale v1 mentre la curva verde al segnale v2. Infine, la curva rossa rappresenta la differenza v1-v2, cioè effettivamente quella che servirà per il calcolo del valore di SNM.

Considerando, pertanto, le definizioni utilizzate per il calcolo di snm\_max e snm\_min all'interno di LTspice, si prende il valore minimo tra i due e moltiplicandolo per  $\frac{1}{\sqrt{2}}$  si ottiene il valore di SNM corrispondente. Bisogna specificare che i valori ottenuti dalla formula .meas specificata per LTspice sono direttamente i valori di SNM poichè è stato considerato il generatore Vsweep per effettuare il calcolo del lato del quadrato corrispondente.

## Comparative Analysis W\_AX-SNM(HOLD)\_MAX-SNM(HOLD)\_MIN Seevinck Method

w_ax [m]	snm(hold)_max_seevinck [mV]	snm(hold)_min_seevinck [mV]
1.2e-07	261.875	261.875
1.3e-07	261.875	261.875
1.4e-07	261.875	261.875
1.5e-07	261.875	261.875
1.6e-07	261.875	261.875
1.7e-07	261.874	261.874
1.8e-07	261.874	261.874
1.9e-07	261.874	261.874
2e-07	261.874	261.874
2.1e-07	261.874	261.874
2.2e-07	261.874	261.874
2.3e-07	261.874	261.874
2.4e-07	261.874	261.87300000000005



Si può notare che i valori generati dalle formule .meas di LTspice rispettivamente per snm\_max e snm\_min sono praticamente gli stessi tranne in corrispondenza dell'ultima w\_ax considerata, cioè per w\_ax = 0.24  $\mu m$ . Pertanto, in questo caso il valore di snm considerato è stato quello di snm\_min essendo quello minimo tra i due. Inoltre, dal momento che la word line risulta essere pari a zero, i cambiamenti di SNM al variare di w\_ax risultano essere praticamente nulli.

### 3.2. Read Phase

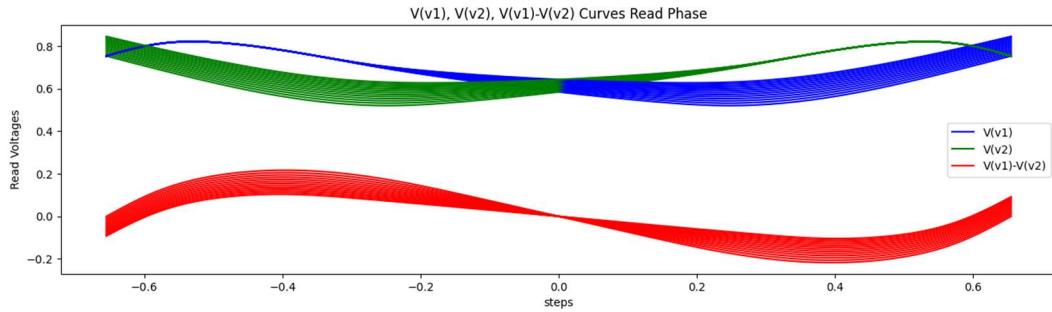
Per l'analisi di Read è stato utilizzato lo script [snm\\_read\\_seevinck\\_analysis.py](#) nel quale è presente il metodo `__init_model__` e dove sono stati specificati il valore SEEVINCK come CircuitType e il valore READ come OperationType. Inoltre, sono state specificate le costanti vdd\_seevinck, vsweep\_seevinck, vwl\_hold, vbl\_hold e vblneg\_hold rispettivamente per i parametri in input vdd, vsweep, vwl, vbl e vblneg.

```

    (
        steps,
        vsweep_seevinck_read,
        i_leaks_seevinck_read,
        v_1_seevinck_read,
        v_2_seevinck_read,
        v_dd_seevinck_read,
        v_wl_seevinck_read,
        v_bl_seevinck_read,
        v_blneg_seevinck_read,
        seevinck_read_log
    ) = __init_model__(
        operation_type=OperationType.READ,
        circuit_type=CircuitType.SEEVINCK,
        asc_file_path=os.path.join(ltspice, "seevinck/read/seevinck_read.asc"),
        schematic_image_path=os.path.join(schematics, "seevinck.png"),
        request_plot_schematic=RequestPlotSchematic.TRUE,
        vdd=vdd_seevinck,
        vsweep=vsweep_seevinck,
        vwl=vwl_read,
        vbl=vbl_read,
        vblneg=vblneg_read,
        params=[rit_models, dc_vsweep_seevinck, snm_max(0.655), snm_min(0.655), w_ax_step_param_seevinck, save_w_ax_seevinck]
)

```

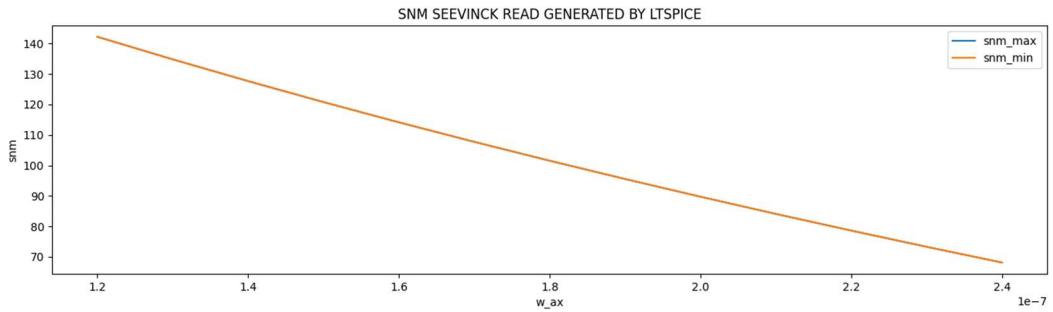
Simulando lo schematico sopra allegato secondo i parametri appena citati otterremo la seguente curva:



Quello che si può notare è che, rispetto alla fase di Hold, le curve generate per ogni  $w_{ax}$  considerata risultano essere differenti poiché la word line risulta essere attiva e, pertanto, il valore di SNM corrispondente ad ogni larghezza di canale del transistor di accesso risulta variare.

#### Comparative Analysis W\_AX-SNM(READ)\_MAX-SNM(READ)\_MIN Seevinck Method

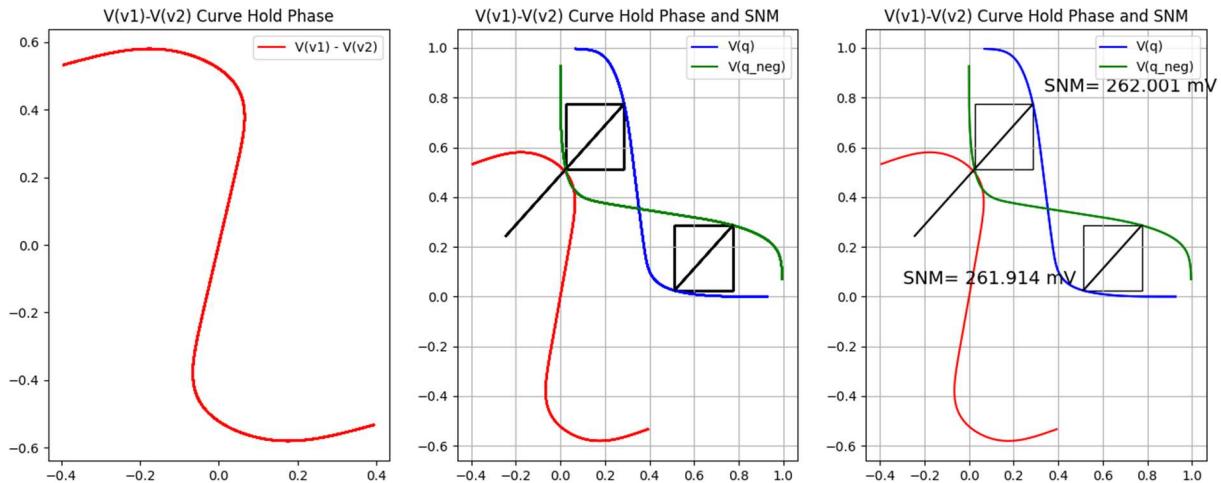
<b>w_ax [m]</b>	<b>snm(read)_max_seevinck [mV]</b>	<b>snm(read)_min_seevinck [mV]</b>
1.2e-07	142.214	142.214
1.3e-07	134.805	134.805
1.4e-07	127.66900000000001	127.66900000000001
1.5e-07	120.78899999999999	120.78899999999999
1.6e-07	114.14699999999999	114.14699999999999
1.7e-07	107.742	107.742
1.8e-07	101.551	101.551
1.9e-07	95.5539	95.5539
2e-07	89.7406	89.7406
2.1e-07	84.10130000000001	84.1012
2.2e-07	78.6269	78.6269
2.3e-07	73.3094	73.3094
2.4e-07	68.14099999999999	68.14099999999999



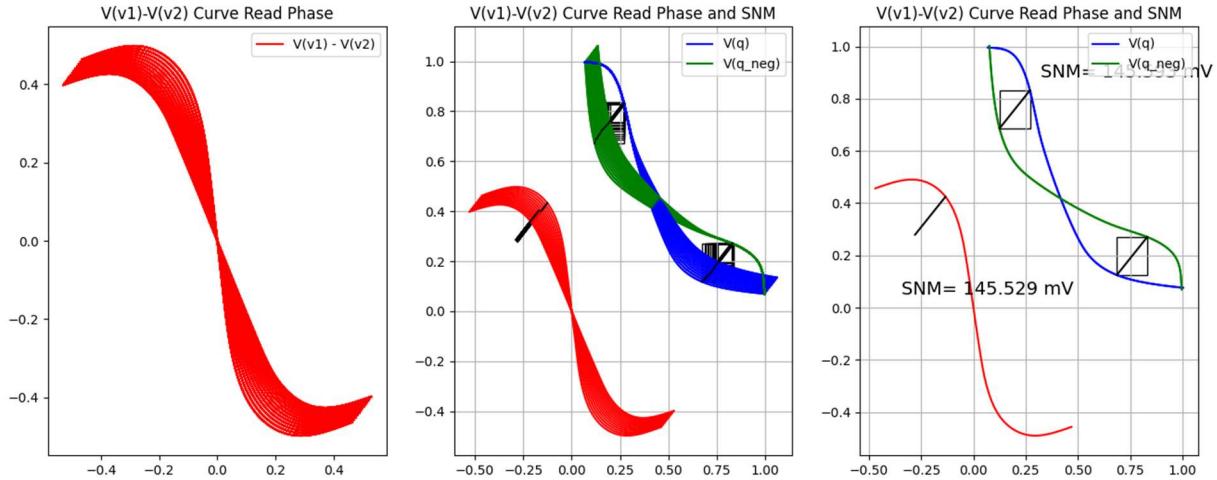
Si può notare, infatti, come il SNM da un valore pari a circa 142 mV in corrispondenza di  $w_{ax} = 0.12 \mu m$  arriva ad un valore pari a circa 68 mV in corrispondenza di  $w_{ax} = 0.24 \mu m$ .

### 3.3. Comparative Analysis between Graphical Method and Seevinck Method

Qui di seguito verranno allegati alcune analisi, ottenute tramite lo script [graphical\\_seevinck\\_comparative\\_analysis.py](#), per effettuare un confronto tra il metodo grafico proposto nel paragrafo 3.1 e il metodo di Seevinck proposto nel paragrafo 3.2.



Dal plot sopra allegato si può notare il confronto tra la curva v1-v2 generata dal metodo di Seevinck e le due curve v1 e v2 a cui viene applicato il metodo grafico. Nel terzo subplot vengono per semplicità rappresentate le curve e i valori corrispondenti di SNM. Si può notare come i valori di SNM risultano essere praticamente uguali tranne per una piccola differenza. Ovviamente nel subplot 1 e nel subplot 2 sono rappresentate tutte le curve ognuna corrispondente al relativo  $w_{ax}$  anche se queste risultano essere praticamente uguali dal momento che la word line è pari a zero trattandosi della fase di Hold.



Dal plot sopra allegato si può notare il confronto tra la curva v1-v2 generata dal metodo di Seevinck e le due curve v1 e v2 a cui viene applicato il metodo grafico. Nel terzo subplot vengono per semplicità rappresentate le curve e i valori corrispondenti di SNM. Si può notare come i valori di SNM risultano essere praticamente uguali tranne per una piccola differenza. Ovviamente nel subplot 1 e nel subplot 2 sono rappresentate tutte le curve ognuna corrispondente al relativo  $w_{ax}$ . Rispetto alla fase di Hold, come già spiegato nei paragrafi precedenti, le curve risultano essere differenti tra loro dal momento che si tratta della fase di Read.

Nelle due successive tabelle e nel plot che le segue verranno rappresentati i valori di SNM, corrispondenti al metodo grafico e al metodo di Seevinck sia per la fase di Hold sia per la fase di Read, per confrontarne le differenze.

Comparative Analysis Graphical-Seevinck

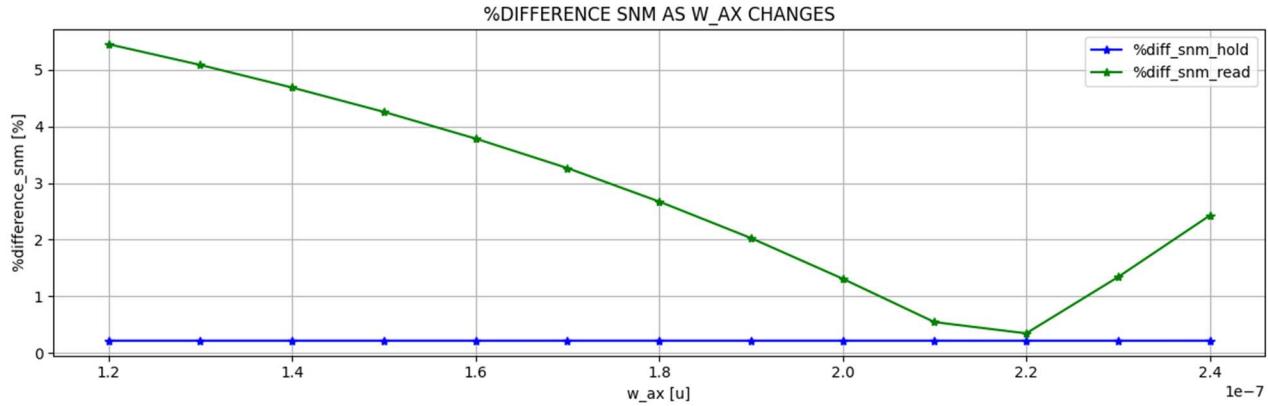
<b>w_ax [m]</b>	<b>snm(hold)_standard [mV]</b>	<b>snm(read)_standard [mV]</b>	<b>snm(hold)_seevinck [mV]</b>	<b>snm(read)_seevinck [mV]</b>
1.2e-07	262.43566244928326	150.18932179736015	261.875	142.214
1.3e-07	262.43550473079256	141.84301463106343	261.875	134.805
1.4e-07	262.43534129914343	133.79837532531718	261.875	127.66900000000001
1.5e-07	262.43517705584117	126.04461719878147	261.875	120.78899999999999
1.6e-07	262.4350036798156	118.55191413438021	261.875	114.14699999999999
1.7e-07	262.43482954124966	111.31813478480773	261.874	107.742
1.8e-07	262.43464673132866	104.30266609752346	261.874	101.551
1.9e-07	262.4344622101111	97.51147039013014	261.874	95.5539
2e-07	262.43426967969657	90.92296793564506	261.874	89.7406
2.1e-07	262.4340759824953	84.55820794965902	261.874	84.1012
2.2e-07	262.43387439480875	78.35772229810932	261.874	78.6269
2.3e-07	262.43367158989145	72.33534210594759	261.874	73.3094
2.4e-07	262.43346020573165	66.5079460850747	261.87300000000005	68.14099999999999

Comparative Analysis %Difference Graphical-Seevinck

<b>w_ax [m]</b>	<b>diff_snm_hold [%]</b>	<b>diff_snm_read [%]</b>
1.2e-07	0.2138665068012	5.4550144973299375
1.3e-07	0.21380640888755423	5.088064442066778
1.4e-07	0.21374413396272512	4.688443686475999
1.5e-07	0.2136815497218566	4.258429024721536
1.6e-07	0.21361548545146747	3.7859344129439707
1.7e-07	0.21393099244212097	3.2649799912893442
1.8e-07	0.2138613333286962	2.673419570016158
1.9e-07	0.21379102208214912	2.0278834947711766
2e-07	0.2137176589027743	1.3089168437836205
2.1e-07	0.21364385106819386	0.5419299821038406
2.2e-07	0.21356703652600098	0.34293512058719045
2.3e-07	0.2134897581011145	1.3375805813076866
2.4e-07	0.2137910738355898	2.4256467835900897

Si può notare come nella fase di Hold i valori di SNM ottenuti tramite metodo grafico e quelli ottenuti dal metodo di Seevinck risultano essere praticamente i medesimi dal momento che la differenza percentuale che si ha è sempre circa pari al 0.214%. Per quanto riguarda, invece, la fase di Read i valori di SNM risultano essere

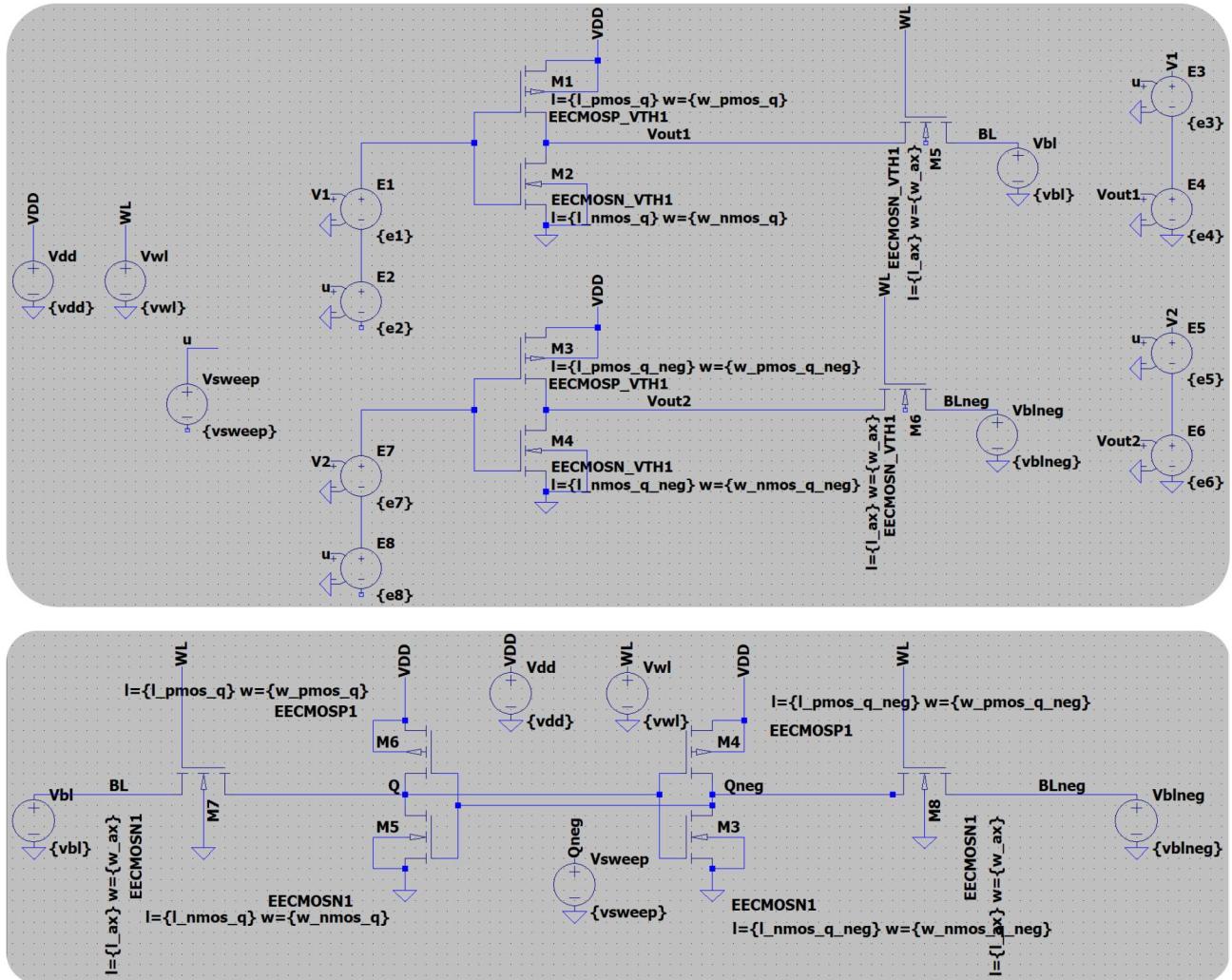
leggermente differenti tra i due metodi utilizzati. In particolare, in corrispondenza di  $w_{ax} = 0.12 \mu m$  si ha una differenza percentuale circa pari al 5.5%. Ovviamente si tratta pur sempre di una piccola differenza ma pur sempre importante perché nel caso del metodo grafico si ottiene un valore di SNM circa pari a 150 mV mentre nel caso del metodo di Seevinck si ottiene un valore di SNM circa pari a 142 mV.



In generale, questo confronto è stato utile per valutare l'accuratezza del metodo grafico: quest'ultimo risulta essere un metodo più semplice da utilizzare dal momento che il circuito utilizzato è quello standard di una cella SRAM 6T ma pur sempre inaccurato poiché l'analisi del valore di SNM avviene graficamente tramite la costruzione di un quadrato inscritto all'interno del lobo più piccolo delle due curve generate. Viceversa, il metodo di Seevinck risulta essere quello più accurato tra i due metodi dal momento che il calcolo del valore di SNM si ottiene tramite il circuito stesso mediante un circuito più complesso ma che permette di analizzare direttamente i segnali di tensione generati dal circuito stesso. Pertanto, la scelta più ovvia sarebbe quella di utilizzare il metodo di Seevinck che permette un'analisi del valore di SNM più accurata anche se il metodo grafico, pur essendo meno accurato, permette di ottenere una stima del valore di SNM accettabile e poco meno accurata del metodo di Seevinck dal momento che il valore di SNM ottenuto al più risulta discostare da quello vero di circa il 0.214% nella fase di Hold e di circa il 5.5% nella fase di Read considerando per entrambi il caso peggiore.

## 4. $V_{DD}$ Scaling of a 6T SRAM considering Gaussian $V_{TH}$

In questo capitolo verrà affrontata l'analisi della cella considerando diversi valori di  $V_{DD}$ , cioè partendo da un valore di  $V_{DD} = 1 V$  fino ad arrivare ad un valore di  $V_{DD} = 0.05 V$  utilizzando uno step di  $0.05 V$ . Pertanto, i passi di scaling considerati saranno 20. Nello specifico, ogni scaling prevederà un numero di simulazioni pari a 10000 dove in ognuna di esse verrà previsto un valore  $V_{TH}$  generato tramite la direttiva 'gauss' prevista per LTspice. In questo modo si riuscirà ad avere un valore accurato di SNM e di corrente di leakage e fare, di conseguenza, analisi a riguardo. Bisogna specificare che per l'analisi del valore di SNM per la fase di Hold e di Read è stata considerata la configurazione del circuito di Seevinck mentre per l'analisi del valore di corrente di leakage è stata considerata la configurazione del circuito utilizzato per l'analisi tramite metodo grafico.



Ovviamente in tutti e tre i casi è stato considerato il modello EECMOSN1 ed EECMOSP1 rispettivamente per il transistor NMOS e per il transistor PMOS così da usufruire della direttiva 'gauss' per la generazione del valore di  $V_{TH}$ .

Ciò che ci si aspetta è che al diminuire della  $V_{DD}$ , il valore di SNM associato alla fase di Hold e quello associato alla fase di Read diminuiscono. Nello specifico, ci sarà un valore di  $V_{DD}$  oltre il quale il valore del margine di rumore crollerà poiché un elevato numero di celle fallirà. Questo però non significa che soltanto oltre quel valore le celle falliranno ma significa, invece, che oltre quel valore di  $V_{DD}$  un gran numero di celle fallirà ma comunque molto probabilmente intorno a quel valore di  $V_{DD}$  un piccolo numero di celle sarà già fallito.

In particolare, la diminuzione del valore di SNM è dovuta al fatto che le curve dalle quali si vengono a creare i lobi si avvicinano sempre di più comportando un restringimento dei lobi sempre maggiore. Di conseguenza,

la diagonale del quadrato e, quindi, la differenza tra v1 e v2 (le due curve considerate) sarà sempre minore avendo, pertanto, un valore sempre minore di SNM.

Per quanto riguarda, invece, l'analisi della corrente di leakage tramite scaling della  $V_{DD}$ , essa risulta essere vantaggiosa per il progettista. Infatti, la diminuzione della potenza di leakage associata alla cella e, pertanto, il decremento associato della corrente di leakage può essere facilmente ottenuto tramite una diminuzione della  $V_{DD}$  considerata per la cella. Infatti, l'obiettivo di questa analisi è ottenere un abbattimento della corrente di leakage associata alla cella.

In generale, questa tipologia di analisi, cioè quella associata allo scaling della  $V_{DD}$ , si propone come obiettivo la ricerca di un valore di  $V_{DD}$  che possa soddisfare sia dal punto di vista del funzionamento della cella, cioè sia il mantenimento del dato che la lettura dello stesso, sia dal punto di vista della potenza di leakage della stessa che deve mantenersi il più basso possibile.

Per il plotting relativo alle distribuzioni di valori ottenuti sia per gli SNM sia per la corrente di leakage verrà utilizzato un sottointervallo di medesima ampiezza per tutte e tre le analisi. Nello specifico, è stato considerato un numero di sotto intervalli per ogni istogramma pari a 100. Pertanto, per ogni sotto intervallo viene calcolato quanti valori di SNM (Hold e Read) e di corrente di leakage ricadono in esso. Per i valori iniziali di  $V_{DD}$  ci si aspetta un trend riconducibile a quello di una distribuzione Gaussiana con la maggioranza dei valori concentrata in corrispondenza della sua media e un raggio dipendente dalla sua deviazione standard. Questo vuol dire che il 49.85% di tali valori sarà contenuto entro il raggio pari a  $\mu - 3\sigma$  (analogamente per  $\mu + 3\sigma$ ), cioè pari alla differenza tra la sua media e la sua deviazione standard.

#### 4.1. Hold Phase

Per l'analisi di Hold è stato utilizzato lo script `snm_hold_vdd_scaling_analysis.py` nel quale è presente il metodo `__init_model__` e dove sono stati specificati il valore GAUSSIAN\_VTH\_DC come `CircuitType` e il valore HOLD come `OperationType`. Inoltre, sono state specificate le costanti `vdd_scaled`, `vsweep_gaussian_vth`, `vwl_hold`, `vdd_scaled` e `vdd_scaled` rispettivamente per i parametri in input `vdd`, `vsweep`, `vwl`, `vbl` e `vblneg`.

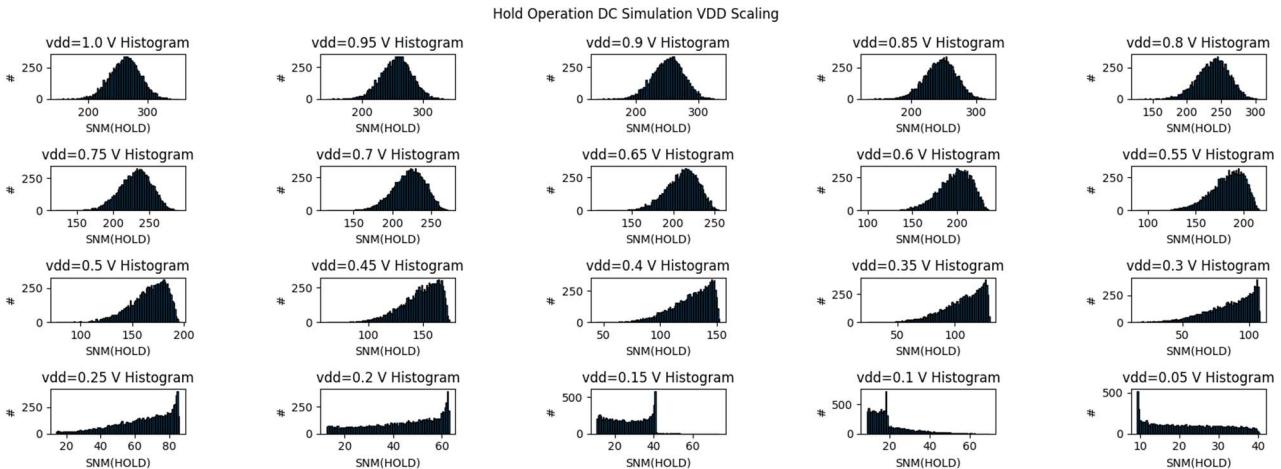
```

for scaling in np.arange(vdd_start, vdd_stop, -vdd_step):
    ...
    vdd_scaled = round(scaling, 2)
    vdd_gaussian_vth_scaled.append(vdd_scaled)
(
    steps,
    vsweep_gaussian_vth_hold,
    i_leaks_gaussian_vth_hold,
    v_1_gaussian_vth_hold,
    v_2_gaussian_vth_hold,
    v_dd_gaussian_vth_hold,
    v_wl_gaussian_vth_hold,
    v_b1_gaussian_vth_hold,
    v_blneg_gaussian_vth_hold,
    gaussian_vth_hold_log
) = __init_model__(
    operation_type=OperationType.HOLD,
    circuit_type=CircuitType.GAUSSIAN_VTH_DC,
    asc_file_path=os.path.join(ltspice, "gaussian-vth/hold/gaussian_vth_hold.asc"),
    schematic_image_path=os.path.join(schematics, "gaussian_vth.png"),
    request_plot_schematic=RequestPlotSchematic.FALSE,
    vdd=f'{vdd_scaled}',
    vsweep=vsweep_gaussian_vth,
    vwl=vwl_hold,
    vbl=f'{vdd_scaled}',
    vblneg=f'{vdd_scaled}',
    params=[
        rit_models_montecarlo,
        dc_vsweep_gaussian_vth(-0.707 * scaling, 0.707 * scaling, 0.01),
        step_param_run_gaussian_vth,
        snm_max(0.707),
        snm_min(0.707)
    ]
)

```

Dallo script sopra allegato si può notare come sia per la  $V_{DD}$  sia per  $V_{bl}$  e  $V_{blneg}$  è stata considerata una  $V_{DD}$  scalata. Tutto ciò implementato tramite un for che considera il range  $[V_{DD,start}, V_{DD,stop}]$  secondo un certo passo come descritto precedentemente.

Eseguendo lo script di plotting corrispondente sarà possibile visualizzare i valori di SNM(HOLD) ottenuti in seguito alle 10000 simulazioni per ogni  $V_{DD}$  considerata. Nello specifico, per ogni  $V_{DD}$  considerata verrà plottato in un subplot un istogramma relativo ai valori di SNM calcolati.



Si può notare come, inizialmente, gli istogrammi assumono il trend di una gaussiana dato dalla direttiva ‘gauss’ utilizzata all’interno di LTspice per la generazione del valore di  $V_{TH}$  e, inoltre, dal numero elevato di simulazioni considerato. Si può osservare che, man mano che la  $V_{DD}$  diminuisce, l’istogramma tende ad assumere un andamento sempre più lontano da quello gaussiano. Nello specifico, il trend risultante per valori sempre

minori di  $V_{DD}$  risulta essere quello di una distribuzione uniforme. Una spiegazione plausibile potrebbe essere quella che dopo un certo valore di  $V_{DD}$  i transistor entrano in regione di conduzione sottosoglia e, pertanto, cambiando la relazione che descrive la corrente, la distribuzione dei valori di SNM risulterebbe non coerente con la distribuzione Gaussiana riscontrata per i valori di  $V_{DD}$  maggiori.

## 4.2. Read Phase

Per l'analisi di Read è stato utilizzato lo script `snm_read_vdd_scaling_analysis.py` nel quale è presente il metodo `__init_model__` e dove sono stati specificati il valore `GAUSSIAN_VTH_DC` come `CircuitType` e il valore `READ` come `OperationType`. Inoltre, sono state specificate le costanti `vdd_scaled`, `vsweep_gaussian_vth`, `vdd_scaled` e `vdd_scaled` rispettivamente per i parametri in input `vdd`, `vsweep`, `vwl`, `vbl` e `vblneg`.

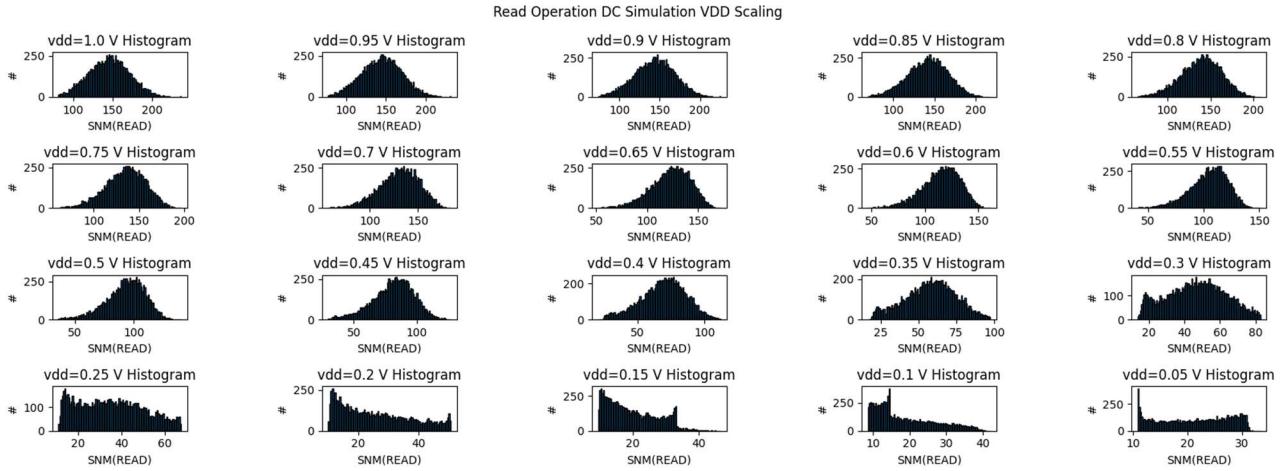
```

for scaling in np.arange(vdd_start, vdd_stop, -vdd_step):
    ...
    vdd_scaled = round(scaling, 2)
    vdd_gaussian_vth_scaled.append(vdd_scaled)
    (
        steps,
        vsweep_gaussian_vth_read,
        i_leaks_gaussian_vth_read,
        v_1_gaussian_vth_read,
        v_2_gaussian_vth_read,
        v_dd_gaussian_vth_read,
        v_wl_gaussian_vth_read,
        v_bl_gaussian_vth_read,
        v_blneg_gaussian_vth_read,
        gaussian_vth_read_log
    ) = __init_model__(
        operation_type=OperationType.READ,
        circuit_type=CircuitType.GAUSSIAN_VTH_DC,
        asc_file_path=os.path.join(ltspice, "gaussian-vth/read/gaussian_vth_read.asc"),
        schematic_image_path=os.path.join(schematics, "gaussian_vth.png"),
        request_plot_schematic=RequestPlotSchematic.FALSE,
        vdd=f'{vdd_scaled}',
        vsweep=vsweep_gaussian_vth,
        vwl=vwl_read,
        vbl=f'{vdd_scaled}',
        vblneg=f'{vdd_scaled}',
        params=[
            rit_models_montecarlo,
            dc_vsweep_gaussian_vth(-0.707 * scaling, 0.707 * scaling, 0.01),
            step_param_run_gaussian_vth,
            snm_max(0.707),
            snm_min(0.707)
        ]
    )

```

Dallo script sopra allegato si può notare come sia per la  $V_{DD}$  sia per  $V_{wl}$ ,  $V_{bl}$  e  $V_{blneg}$  è stata considerata una  $V_{DD}$  scalata. Tutto ciò implementato tramite un `for` che considera il range  $[V_{DD,start}, V_{DD,stop}]$  secondo un certo passo come descritto precedentemente.

Eseguendo lo script di plotting corrispondente sarà possibile visualizzare i valori di SNM(READ) ottenuti in seguito alle 10000 simulazioni per ogni  $V_{DD}$  considerata. Nello specifico, per ogni  $V_{DD}$  considerata verrà plottato in un subplot un istogramma relativo ai valori di SNM calcolati.



Si potrebbe fare un ragionamento analogo riguardo la distribuzione Gaussiana che si ha fino ad un certo valore di  $V_{DD}$  e il cambiamento di andamento che si ha per valori sempre minori di  $V_{DD}$  come detto nel paragrafo precedente.

#### 4.3. $I_{LEAK}$ Analysis

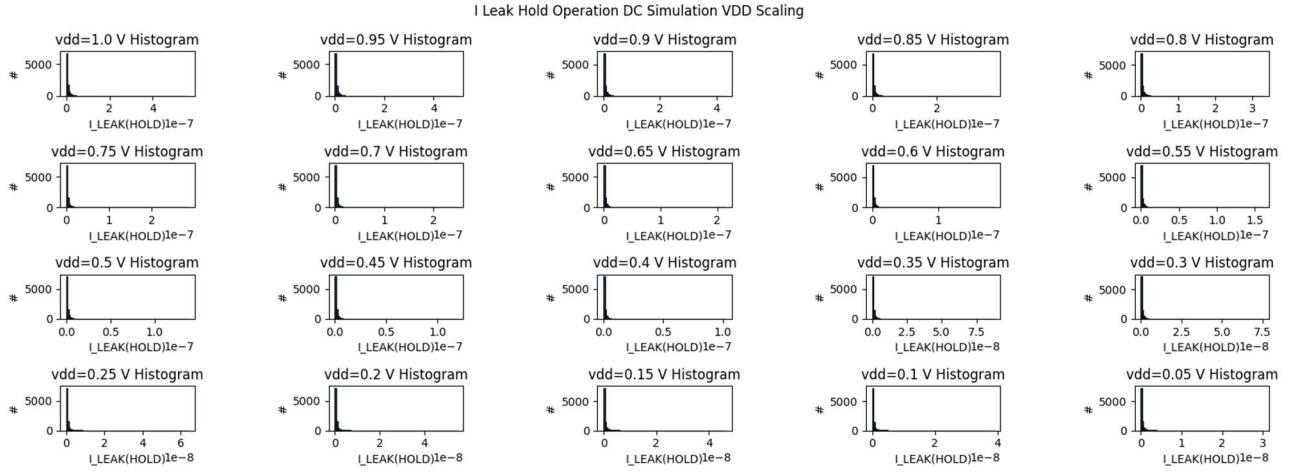
Per l'analisi della corrente di leakage è stato utilizzato lo script `i leak_vdd_scaling_analysis.py` nel quale è presente il metodo `__init__model__` e dove sono stati specificati il valore `STANDARD_ILEAK` come `CircuitType` e il valore `HOLD` come `OperationType`. Inoltre, sono state specificate le costanti `vdd_scaled`, `vsweep_standard_ileak`, `vwl_hold`, `vdd_scaled` e `vdd_scaled` rispettivamente per i parametri in input `vdd`, `vsweep`, `vwl`, `vbl` e `vblneg`.

```
for scaling in np.arange(vdd_start, vdd_stop, -vdd_step):
    ...
    vdd_scaled = round(scaling, 2)
    vdd_standard_dc_scaled.append(vdd_scaled)
    (
        steps,
        time_standard_hold,
        i_leaks_standard_hold,
        v_q_standard_hold,
        v_q_neg_standard_hold,
        v_dd_standard_hold,
        v_wl_standard_hold,
        v_bl_standard_hold,
        v_blneg_standard_hold,
        standard_hold_log
    ) = __init__model__(
        operation_type=OperationType.HOLD,
        circuit_type=CircuitType.STANDARD_ILEAK,
        asc_file_path=os.path.join(ltspice, "standard/hold/i leak/standard_hold_i leak.asc"),
        schematic_image_path=os.path.join(schematics, "standard_i leak.png"),
        request_plot_schematic=RequestPlotSchematic.FALSE,
        vdd=f'{vdd_scaled}',
        vsweep=vsweep_standard_ileak,
        vwl=vwl_hold,
        vbl=f'{vdd_scaled}',
        vblneg=f'{vdd_scaled}',
        params=[
            rit_models_montecarlo,
            step_param_run_standard_ileak,
            dc_vsweep_scaling(vdd_scaled)
        ]
    )

```

Dallo script sopra allegato si può notare come sia per la  $V_{DD}$  sia per  $V_{bl}$  e  $V_{blneg}$  è stata considerata una  $V_{DD}$  scalata. Tutto ciò implementato tramite un for che considera il range  $[V_{DD,start}, V_{DD,stop}]$  secondo un certo passo come descritto precedentemente.

Eseguendo lo script di plotting corrispondente sarà possibile visualizzare i valori di corrente di leakage ottenuti in seguito alle 10000 simulazioni per ogni  $V_{DD}$  considerata. Nello specifico, per ogni  $V_{DD}$  considerata verrà plottato in un subplot un istogramma relativo ai valori di corrente di leakage calcolati.



#### 4.4. Cell Robustness and Leakage Power

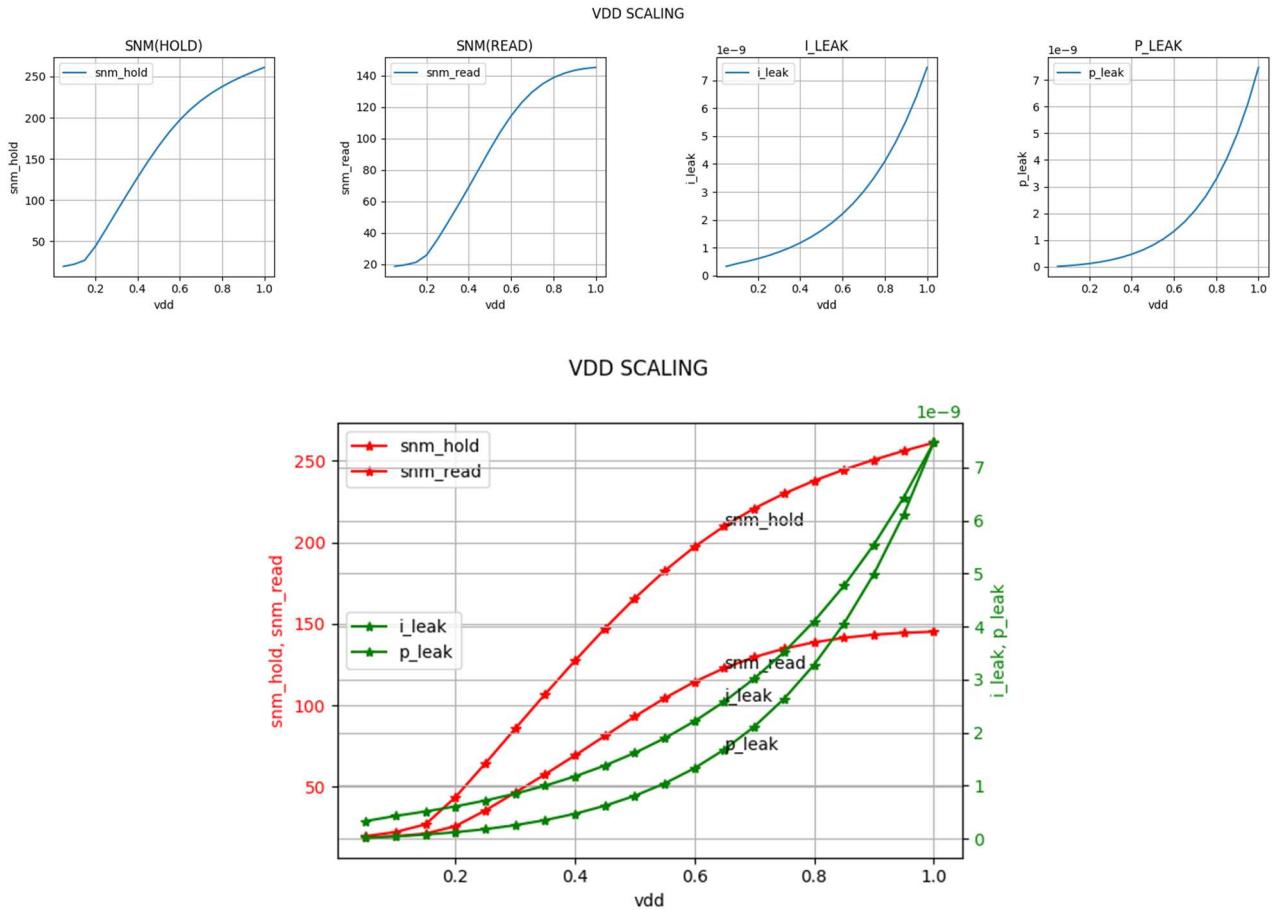
Si analizzano, pertanto, i risultati complessivi ottenuti dal  $V_{DD}$  scaling per l'SNM e per la corrente di leakage per trovare un buon compromesso rispettivamente tra la robustezza della cella e la dissipazione di potenza statica.

Comparative Analysis VDD Scaling

vdd [V]	snm(hold)_mean [mV]	snm(hold)_stdev [mV]	snm(read)_mean [mV]	snm(read)_stdev [mV]	ileak(hold)_mean [A]	ileak(hold)_stdev [A]	pleak(hold)_mean [W]	pleak(hold)_stdev [W]
1.0	261.04257	25.55507	145.10223	25.45446	7.46e-09	1.568e-08	7.46e-09	1.568e-08
0.95	256.0513	25.10047	144.43667	25.19701	6.43e-09	1.361e-08	6.11e-09	1.293e-08
0.9	250.58968	24.54336	143.24933	24.76373	5.54e-09	1.138e-08	4.99e-09	1.062e-08
0.85	244.53134	23.85896	141.33973	24.1124	4.77e-09	1.022e-08	4.05e-09	8.69e-09
0.8	237.70105	23.02309	138.58739	23.25802	4.1e-09	8.84e-09	3.38e-09	7.07e-09
0.75	229.86398	22.02927	134.71961	22.16388	3.52e-09	7.64e-09	2.64e-09	5.73e-09
0.7	220.71969	20.90343	129.53146	20.87102	3.02e-09	6.59e-09	2.11e-09	4.61e-09
0.65	209.72564	19.73509	122.77771	19.45894	2.59e-09	5.68e-09	1.68e-09	3.69e-09
0.6	197.19545	18.68933	114.33841	18.09961	2.21e-09	4.89e-09	1.33e-09	2.93e-09
0.55	182.38782	17.9579	104.30747	17.00802	1.89e-09	4.2e-09	1.04e-09	2.31e-09
0.5	165.55851	17.97516	93.03773	16.37804	1.62e-09	3.6e-09	8.1e-10	1.8e-09
0.45	147.06055	17.51706	81.10806	16.26739	1.38e-09	3.08e-09	6.2e-10	1.39e-09
0.4	127.28477	17.53982	69.1549	16.51236	1.17e-09	2.64e-09	4.7e-10	1.06e-09
0.35	106.69758	17.58807	57.56758	16.71094	1e-09	2.25e-09	3.5e-10	7.9e-10
0.3	85.58993	17.52509	46.33672	16.36782	8.5e-10	1.92e-09	2.5e-10	5.8e-10
0.25	64.26217	17.10206	35.42107	14.90359	7.2e-10	1.63e-09	1.8e-10	4.1e-10
0.2	43.56414	14.84075	25.77129	11.64003	6.1e-10	1.38e-09	1.2e-10	2.8e-10
0.15	26.8792	9.8002	21.19308	6.37025	5.1e-10	1.17e-09	8e-11	1.8e-10
0.1	22.1789	9.16865	19.6427	7.50722	4.3e-10	9.8e-10	4e-11	1e-10
0.05	19.62108	10.4708	18.63857	8.07109	3.3e-10	7.6e-10	2e-11	4e-11

Dalla tabella sopra allegata si possono analizzare nel dettaglio i valori relativi agli SNM per la fase di Hold e per la fase di Read e i valori di corrente di leakage e potenza di leakage associati alla fase di Hold. Non considerando il numero di celle che potrebbero fallire in corrispondenza di un certo valore di  $V_{DD}$ , presumibilmente, per la cella in questione, si potrebbe considerare un valore di  $V_{DD}$  pari a 0.35 V poiché si riuscirebbe a garantire un grado di robustezza minimo della cella dal momento che sia l'SNM di Hold sia l'SNM di Read risultano essere circa maggiori di 60 mV e, inoltre, si riuscirebbe ad avere un sostanzioso abbattimento della potenza di leakage dal momento che quest'ultima risulterebbe essere diminuita di un ordine di grandezza.

Nei successivi plot si può osservare nel dettaglio il trend di ogni grandezza considerata durante tale analisi.



#### 4.5. DRV Analysis

Il DRV rappresenta la tensione minima oltre la quale si avrebbe il fallimento della cella di memoria. Nello specifico, esso rappresenta un limite per lo scaling della  $V_{DD}$  riguardo la robustezza della cella. In particolare, esso delinea la tensione minima a cui corrisponde un margine di rumore in fase di Hold pari ad una determinata soglia di robustezza che in questo caso è stata scelta pari a circa 60 mV. Il calcolo del DRV, quindi, consiste nel calcolare, per ogni valore di  $V_{DD}$  considerato durante lo scaling, quante celle di memoria falliscono, cioè quante celle presentano un margine di rumore in fase di Hold uguale a 60 mV.

Eseguendo lo script [\*drv\\_analysis.py\*](#) è possibile effettuare l'analisi riguardo il DRV della cella in questione. Bisogna specificare che la soglia considerata è circa 60 mV, cioè è stato considerato un range di soglia pari a (58.5,61.5), cioè estremi esclusi, dal momento che, essendo valori decimali ed essendo che le analisi sono state effettuate tramite diversi software, le approssimazioni riguardo questi valori potrebbero essere grossolane. Pertanto, avere un valore di margine di rumore con diverse cifre decimali pari a 59 o 61 risulta essere la stessa cosa che avere un margine di rumore perfettamente preciso a 60 mV.

```

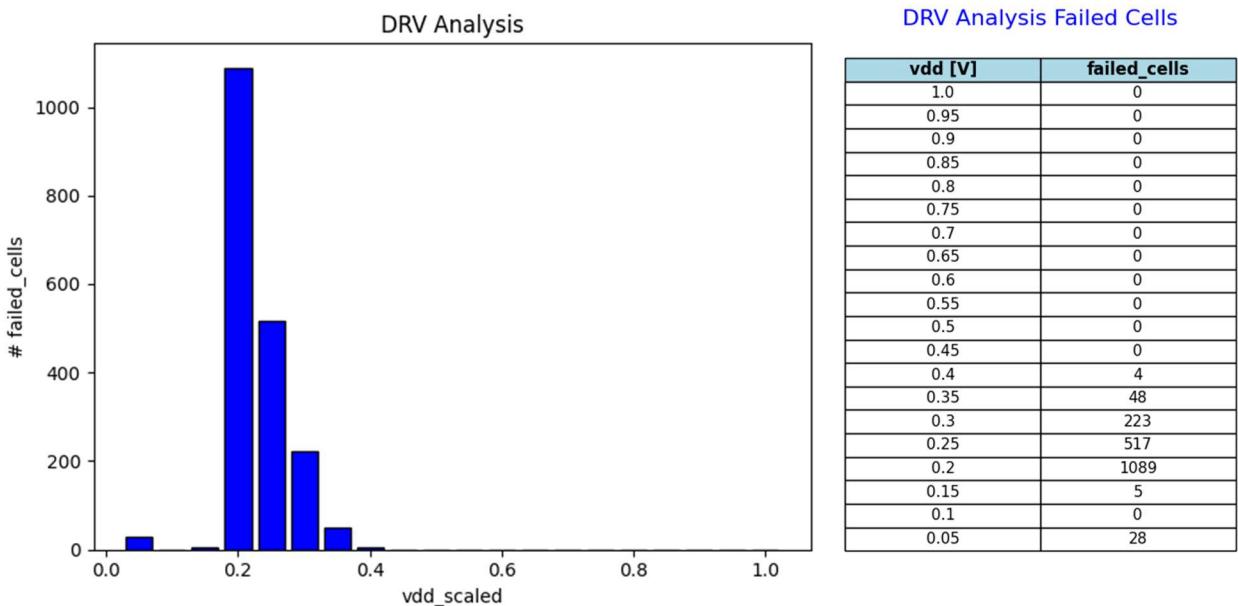
range_threshold = [58.5, 61.5]

counter = []

for row in snm_hold_values:
    count = 0
    for val in row:
        if range_threshold[0] < val < range_threshold[1]:
            count = count + 1
    counter.append(count)

```

Pertanto, dalla seguente analisi si ottiene un istogramma rappresentante il numero di celle che falliscono per ogni  $V_{DD}$  considerata durante lo scaling.



Il plot sopra allegato mostra l'andamento delle celle fallite al variare dello scaling della  $V_{DD}$ . In particolare, il trend, ottenuto dall'analisi del DRV sopra citata, risulta essere quello previsto, cioè quello di una funzione log-normale.

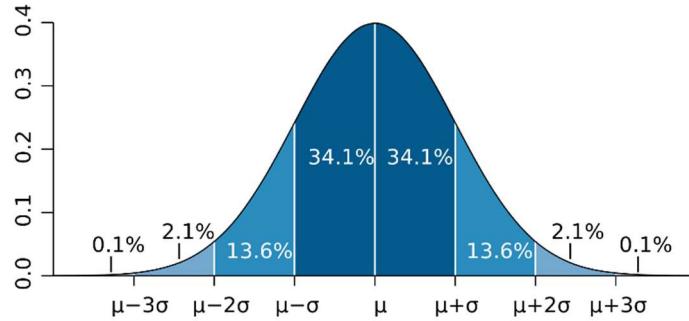
Per quanto riguarda il numero delle celle fallite, si può notare come in corrispondenza di  $V_{DD} = 0.4V$  si ha il fallimento della cella. Questo vuol dire che la tensione minima per garantire la robustezza della cella è  $V_{DD} = 0.45V$ . Pertanto, il trade-off citato nel precedente paragrafo bisognerebbe modularlo ai risultati appena ottenuti. Quindi, il compromesso che si potrebbe raggiungere per avere sia una buona robustezza della cella sia per avere una ridotta dissipazione di potenza statica si ha in corrispondenza di  $V_{DD} = 0.45V$ . Nello specifico, in corrispondenza di tale tensione di alimentazione la cella di memoria presenta i seguenti parametri:

$V_{DD}$	SNM(HOLD) MEAN	SNM(HOLD) STDEV	SNM(READ) MEAN	SNM(READ) STDEV	I <sub>LEAK</sub> MEAN	I <sub>LEAK</sub> STDEV	P <sub>LEAK</sub> MEAN	P <sub>LEAK</sub> STDEV	FAILED CELLS
0.45 V	147.06055 mV	17.51706 mV	81.10806 mV	16.26739 mV	1.38e <sup>-09</sup>	3.08e <sup>-09</sup>	6.2e <sup>-10</sup>	1.39e <sup>-09</sup>	0

Si può notare come in corrispondenza di  $V_{DD} = 0.45V$  si riesce ad ottenere un buon compromesso tra la robustezza della cella e la dissipazione di potenza statica associata. Infatti, si può notare come il valore di margine di rumore per la fase di Hold sia abbondantemente sopra la soglia minima dei 60 mV ma anche il

valore di SNM per la fase di Read risulta essere al di sopra della soglia appena citata. Inoltre, il valore di corrente di leakage e, pertanto, di potenza di leakage corrispondente risultano essere ridotti di un ordine di grandezza rispetto ai valori ottenuti per  $V_{DD}$  maggiori.

Bisogna notare che, trattandosi di distribuzioni gaussiane, il 99.7% dei valori corrispondenti alla distribuzione sono contenuti entro il raggio sinistro pari a  $\mu - 3\sigma$  e il raggio destro pari a  $\mu + 3\sigma$ . Pertanto, affinché venga fatta un'analisi accurata del trade-off possibile, bisogna considerare quest'ultimo aspetto citato.



Questo concetto è molto importante per ciò che riguarda la robustezza della cella poiché, ad esempio, considerando la configurazione scelta come trade-off, cioè quella per  $V_{DD} = 0.45V$ , si ha che il 99.7% dei valori di SNM in fase di Hold sono contenuti in un range pari a  $[147.06055 - 3 \cdot 17.51706, 147.06055 + 3 \cdot 17.51706] = [147.06055 - 52.55118, 147.06055 + 52.55118] = [94.50937, 199.61173] mV$  che risulta essere accettabile per la soglia di robustezza scelta pari a 60 mV. Invece, nel caso in cui si considerasse una soglia per la valutazione della correttezza dell'operazione di lettura per la cella considerata, bisognerebbe fare un ragionamento analogo per la scelta del giusto trade-off senza rischiare di scegliere una  $V_{DD}$  che non possa però garantire una corretta lettura del dato. Infatti, considerando il compromesso scelto in corrispondenza di  $V_{DD} = 0.45V$ , si può notare che il 99.7% di valori di SNM per la fase di Read risulta essere compreso nell'intervallo  $[81.10806 - 3 \cdot 16.26739, 81.10806 + 3 \cdot 16.26739] = [81.10806 - 48.80217, 81.10806 + 48.80217] = [32.30589, 129.91023] mV$ . Questo vuol dire che la quasi totalità dei valori di SNM per la fase di Read sarà compresa in un range dove i valori possibili potrebbero essere anche minori della possibile soglia prestabilita pari a 60 mV. Pertanto, la configurazione scelta come trade-off potrebbe comportare, per quanto riguarda l'operazione di lettura, un'operazione di Read non corretta per alcune determinate celle. Ovviamente, in questo caso per semplicità, è stato considerato come parametro di riferimento soltanto il DRV e, pertanto, è stato considerato come parametro di soglia la robustezza della cella, cioè il corretto mantenimento del dato nella cella di memoria di riferimento. Per maggiori dettagli bisogna fare riferimento ai valori di SNM e ai corrispondenti range elencati nella seguente tabella:

Comparative Analysis VDD Scaling with DRV Analysis										
vdd [V]	snm(holder)_mean [mV]	snm(holder)_stdev [mV]	snm(read)_mean [mV]	snm(read)_stdev [mV]	ileak(holder)_mean [A]	ileak(holder)_stdev [A]	pleak(holder)_mean [W]	pleak(holder)_stdev [W]	failed_cells	incorrect_readings
1.0	261.04257	25.55507	145.10223	25.45446	7.46e-09	1.56e-08	3.7e-10	7.8e-10	0	0
0.95	256.0513	25.10047	144.43667	25.19701	6.43e-09	1.36e-08	6.4e-10	1.36e-09	0	0
0.9	250.58968	24.54336	143.24933	24.76373	5.54e-09	1.18e-08	8.3e-10	1.77e-09	0	0
0.85	241.3134	23.58996	141.93773	24.1124	4.67e-09	9.52e-09	1.55e-10	2.04e-09	0	0
0.8	237.79525	23.0526	139.58739	23.52602	4.1e-09	8.85e-09	1.4e-10	2.02e-09	0	0
0.75	229.88398	22.69227	134.71961	22.16388	3.52e-09	7.64e-09	1.06e-09	2.20e-09	0	2
0.7	220.71969	20.90243	129.53146	20.87102	3.02e-09	6.59e-09	1.06e-09	2.31e-09	0	13
0.65	209.92564	19.73509	122.77271	19.45894	2.59e-09	5.68e-09	1.03e-09	2.27e-09	0	9
0.6	197.19545	18.68913	114.33841	18.09961	2.21e-09	4.89e-09	1e-09	2.2e-09	0	16
0.55	182.38782	17.9579	104.30747	17.00802	1.89e-09	4.2e-09	9.5e-10	2.1e-09	0	49
0.5	165.55851	17.57516	93.03773	16.37804	1.62e-09	3.6e-09	8.9e-10	1.98e-09	0	101
0.45	147.06055	17.51706	81.10806	16.26739	1.38e-09	3.08e-09	8.3e-10	1.85e-09	0	270
0.4	127.28477	17.53982	69.1549	16.51236	1.17e-09	2.64e-09	7.6e-10	1.71e-09	4	549
0.35	106.69758	17.58807	57.56758	16.71094	1e-09	2.25e-09	7e-10	1.58e-09	48	683
0.3	85.58633	17.72509	46.33677	16.36782	8.5e-10	1.92e-09	6.4e-10	1.44e-09	223	537
0.25	64.64332	17.8626	35.53507	16.24502	5.3e-10	1.55e-09	5.3e-10	1.35e-09	517	535
0.2	43.56414	14.84075	25.77129	11.64003	6.1e-10	1.38e-09	5.2e-10	1.18e-09	1089	0
0.15	26.8792	9.8002	21.19308	6.37025	5.1e-10	1.17e-09	4.6e-10	1.05e-09	5	0
0.1	22.1789	9.16865	19.6427	7.50722	4.3e-10	9.6e-10	4.1e-10	9.3e-10	0	0
0.05	19.62108	10.34708	18.63857	8.07109	3.3e-10	7.6e-10	3.3e-10	7.6e-10	28	0