UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

COMME EXIGENCE PARTIELLE
DE LAMAÎTRISE EN GÉNIE ÉLECTRIQUE

PAR
BAHMAN GHOLAMI

APPLICATION DES SYSTÈMES DE CALCUL À HAUTE PERFORMANCE DANS LES ÉTUDES ÉLECTROTHERMIQUES À L'ÉCHELLE NANOSCOPIQUE

AVRIL 2011

Résumé

La miniaturisation des MOSFETs a été le carburant de l'industrie semi-conducteur pour quelque dizaine d'années. L'industrie continuera, probablement, son développement à l'avenir en créant de nouveaux concepts et technologies. À long terme, pour arriver à une miniaturisation extrême en comparaison avec l'état de l'art, de nouveaux concepts de transistors sont prévus pour remplacer les MOSFETs conventionnelles. Même dans la prochaine décennie, des corrections dans la modélisation et conception des MOSFETs sont prévues. Au même temps que la miniaturisation arrive aux barrières physiques, de nouveaux phénomènes voient le jour. L'un de ces phénomènes est le transport balistique.

Le transport balistique et les autres phénomènes reliés à physique quantique, qui devienne dominant quand les tailles de composant atteignent un certain niveau, ont besoin d'algorithmes de calcul très lourds pour être modélisés avec précision. L'utilisation des techniques de calcul à haute performance (HPC) est une solution prometteuse pour le problème de modélisation et conception de MOSFETs du futur.

Dans ce travail, un outil de modélisation créé par les chercheurs à l'université de Perdue est employé comme un algorithme qui modélise de nouveaux effets de transport quantique dans les MOSFETs à double grille. Cet algorithme est parallélisé avec MATLAB et exécuté sur une infrastructure HPC de consortium CLUMEQ, sous la plate-forme Calcul Canada.

Abstract

MOSFET scaling has been the fuel of semiconductor industry for decades. The industry, most probably, will continue its development in the future; but for that to happen, new concepts and technologies have to be created along the way, as it has been the case in the past decades. In the long run, to achieve extreme scaling in comparison to the state of the art, new concepts of transistors are expected to replace conventional MOSFETs; even in the next decade, corrections to MOSFET modeling and design are expected to be necessary as scaling reaches some physical barriers and exposes new phenomena, one of which is ballistic transport.

Ballistic transport and other quantum physics related phenomena, that become dominant when device sizes reach certain levels, need very computationally expensive algorithms to be modeled precisely. The use of high performance computing techniques is a hopeful solution to the problem of modeling and designing future MOSFETs.

In this work, a modeling tool created by researchers at Purdue University is employed as an algorithm that models new quantum transport effects in double gate MOSFETs. This algorithm is parallelized under MATLAB and run on HPC infrastructure consortiums CLUMEQ and SHARCNET, under Compute Canada platform.

Dedication

To my parents whose devotion and sacrifice is endless.

To Azar, with love.

Acknowledgments

I thank Professor Adam W. Skorek for great motivation and courage that he gave me to start this project and for his continued support and trust throughout my academic program at UQTR.

I also thank Simon Delisle for his kind and valuable help.

I would also like to thank Vladimir Timochevski from CLUMEQ for his effective help.

Contents

List of tables

List of figures

List of symbols

C	Capacitance
C_{ox}	Oxide capacitance
D_{Si} , D_{Di}	effects of source and drain on the local density function for subband i
D_{ji}	Local density
$\vec{E}$	Electric field
E_i	Bound state energy of sub band i
$E_l = E - E_{k_j}$	Longitudinal energy
E_{pt}	Maximum energy of subband i
f	frequency
F_n	Quasi-Fermi energy
G	Green's function
G^+	Hermitian conjugate of G
G^n , G^p	Correlation functions describing electron and hole density
G_m^n	mth diagonal entry of G^n
h	Plank constant
H	Single-electron effective mass Hamiltonian

I	Current
I_d	Drain current
$I_m(E)$	Current spectrum at terminal m
I_{Oi}	Constant
I_{ej}	Current in sources of scattering or carrier
J	Current density
k_B	Constant
L	Inductance
m	Mass
m_z^*	Electron effective mass in z direction
$< m^* >$	Average of electron effective mass in the transport direction
n	Electron density
$n(E,m)$	Electron density spectrum
N_D, N_A	Donor and acceptor's doping concentrations
N_x, N_z	Grid node numbers in x and z directions
N_c	Effective density of states in the conduction band
n_{2D_i}	Electron density constant
n_{Oi}	Electron density constant with scattering

p	Momentum
p	Hole density
P	Parallelizable percentage of algorithm
P	Number of processors
p_{2D_i}	Constant
q	Elementary charge constant
$q < \tau >$	Average of electron state lifetime
Q	Charge
r	Space vector
R_{on}	Channel on resistance
S	Speed-up factor
t	Time
$T_{SDi}(E)$	Source-to drain transition in terms of electron energy
T_{jki}	Transmission from k to j in subband i
v	Speed of electron
V_d	Drain voltage
V_{dd}	Operating voltage
V_{DS}	Drain-source voltage

V_g	Gate voltage
V_{GS}	Gate-source voltage
V_t, V_{th}	Threshold voltage
Vsat	Saturation velocity
W	Energy
α	Non-parallelizable part of algorithm
Beta	Caughy-Thomas parameter
ε	Dielectric constant
μ	Body Fermi energy
mu_low	Low field mobility
μ_n	Mobility of electron
μ_S, μ_D	Contact Fermi energies of source and drain
μ_j	Fermi potential
Σ	Self-energy matrix
Σ, Σ^{in} and Σ^{out}	Self-energy functions
Σ^{in}_m mth	Diagonal entry of Σ^{in}
$\Psi(r)$	Field operator
$\Psi_i(z)$	Envelope function of subband i

$\mathfrak{I}_{1/2}$ Fermi-Dirac integral of order ½

$\mathfrak{I}_{-1/2}$ Fermi integral of order -1/2

Résumé français

Introduction

La miniaturisation des MOSFETs a été le carburant de l'industrie semi-conducteur pour quelque dizaine d'années. L'industrie continuera, probablement, son développement à l'avenir en créant de nouveaux concepts et technologies. À long terme, pour arriver à une miniaturisation extrême en comparaison avec l'état de l'art, de nouveaux concepts de transistors sont prévus pour remplacer les MOSFETs conventionnelles. Même dans la prochaine décennie, des corrections dans la modélisation et conception des MOSFETs sont prévues. Au même temps que la miniaturisation arrive aux barrières physiques, de nouveaux phénomènes voient le jour. L'un de ces phénomènes est le transport balistique.

Le transport balistique et les autres phénomènes reliés à physique quantique, qui devienne dominant quand les tailles de composant atteignent un certain niveau, ont besoin d'algorithmes de calcul très lourds pour être modélisés avec précision. L'utilisation des techniques de calcul à haute performance (HPC) est une solution prometteuse pour le problème de modélisation et conception de MOSFETs de l'avenir.

Dans ce travail, un outil de modélisation créé par les chercheurs à l'université de Perdue est employé comme un algorithme qui modélise de nouveaux effets de transport quantique dans les MOSFETs à double grille. Cet algorithme est parallélisé avec MATLAB et exécuté sur une infrastructure HPC de consortium CLUMEQ, sous la plate-forme Calcul

Canada.

Objectif

Aujourd'hui, la miniaturisation des MOSFETs envisage un obstacle majeur qui est la consommation de puissance. Dans les technologies courent, la miniaturisation augment la densité de puissance (figure 1), la génération de chaleur et la température de chip à un niveau très élevé dans le quel la fonctionnalité et la fiabilité de composant sont menacés [5].

Densité de chaleur élevée et génération non-uniforme de chaleur, qui est présent dans les nano-MOSFETs, créent les gradients forts de température [4].

Température élevée pourrais créer des problèmes variés: décharge électrostatique, électro-migration, surcharge électrique, courent de perte, erreur de synchronisation, auto-échauffement, point chauds, stress thermique, etc. [4]

Propriétés de transfert de chaleur dans l'échelle macroscopique ne sont pas identiques avec ces propriétés dans l'échelle nanoscopique. Conduction de chaleur dans l'échelle nanoscopique est balistique et similaire à radiation thermique [6].

Dans les échelles très petits, « points chauds » de l'échelle nanoscopique, créés dans la région drain de transistor, augment résistance série de drain et résistance d'injection de source. Dans les géométries non-conventionnelles qui utilisent nouvelles matériaux (par exemple dans transistor ultra-mince, finFET ou nanofil) ce phénomène est même plus fort. En plus, augmentation de ratio de surface à volume dans les nouvelles géométries (figure 2), augment la résistance thermique de frontière des matériaux. Avec les nouvelles géométries il est plus difficile de faire sortir la chaleur créée à l'intérieur de composant.

Nouvelles matériaux utilisés dans ces nouveaux composants ont les conductivités thermiques beaucoup plus faible que les matériaux utilisés dans les composants bulks (Table 1) [5].

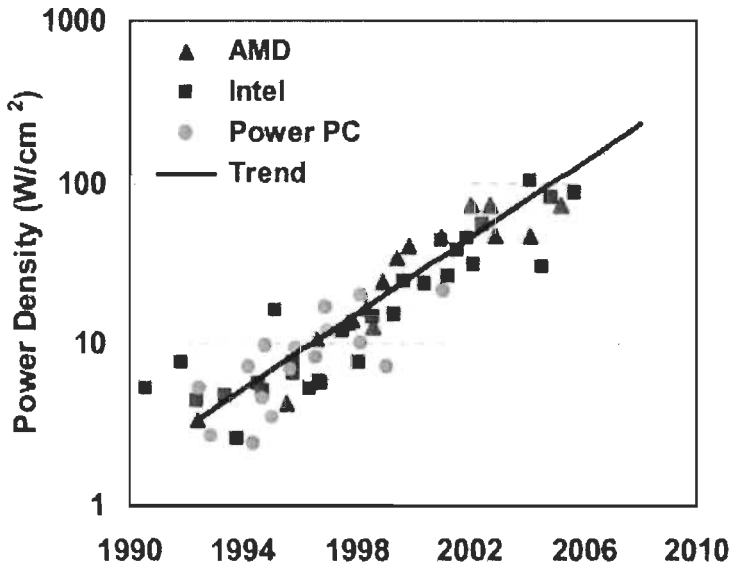


Figure 1 Densité de puissance dans le temps ([9])

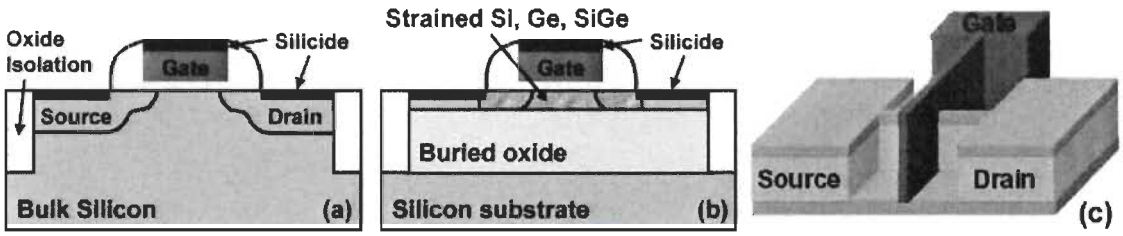


Figure 2 (a) Silicium bulk, (b) Silicium/Germanium tendu sur isolant, (c) composants multi-grille ou FinFET. Conductivité thermique dans technologies plus avancés (b et c) est plus faible. (c) est pris de ITRS ([9])

Table 1 Conductivités thermiques des matériaux utilisés dans la fabrication de semi-conducteurs ([5])

Material	Thermal Conductivity (W/m*K)
Si	148
Ge	60
Silicide	40
Si (10 nm)	13
SiO_2	1.4

En plus, dans les composant où l'une ou plus d'une des dimensions de semi-conducteur de canal est plus petite que « mean free path » d'électron ou phonon, le déplacement des

transporteurs est décrit par les modèles de transport balistique. Transport balistique des particules crée les points chauds et cela change la modèle thermique de composant. Dans ces tailles, où les films semi-conducteurs sont plus minces que « mean free path », conductivité thermique de ces films est diminuée à cause de confinement de phonon et diffusion de frontières [5].

Avant arriver à ce niveau de miniaturisation, les points chauds apparaissait qu'à dans le niveau de circuit et ils étaient reliés à l'architecture de chip. Mais maintenant, avec miniaturisation extrême, les points chauds nanoscopique sont créés à l'intérieur de transistors [9].

Miniaturisation conventionnelle de CMOS se compose de réduire la longueur de grille et la largeur de diélectrique de grille et augmenter dopage du canal. Dans les MOSFETs plane, dopage élevé du canal diminue les effets de canal court, mais il aussi réduit la mobilité et augment courent de perte. La fonction principale de solutions multi-grille et « fully-depleted » ultra-mince est de diminuer ces effets indésirables et améliorer la performance.

MOSFETs à double grilles sont les composants avec deux électrodes de grille, placés à deux côtés opposés du canal. Cette structure donne un meilleur control sur canal. Ces deux électrodes de grille peuvent être deux surfaces d'une grille ou deux électrodes indépendantes, commandés par deux signaux de grille [20].

Présentement, la demande de calcul élevée d'analyse électrothermique nanoscopique cause des difficultés dans l'optimisation de conception. Pour arriver aux conceptions efficientes de circuits basés sur nano transistors, cet obstacle doit être battu [15].

Technique rapide d'analyse thermique, basée sur transport classique Fourier, peut servir pour optimiser les conceptions de chips et de refroidissement. Mais cette technique est incapable de décrire les effets nanoscopiques, parce qu'elle manque la description quantique de phonons. D'autres techniques sont développées pour modéliser ces effets : méthodes dynamiques moléculaires, équation de transport de Boltzmann et modèle de diffusion balistique. Mais complexité de calcul reste d'être un obstacle. Il y a beaucoup de différence entre les modèles d'analyse thermique nanoscopique de chip, optimisées pour efficacité et les modèles qui sont optimisées pour précision [14].

Une solution pour le problème de simulation de modèles thermique exigeantes est d'employer les algorithmes parallèles et les implémenter dans les systèmes de calcul à haut performance [15].

Quand la miniaturisation des MOSFETs entre l'échelle nanoscopique, nouvelles effets de nature quantique deviendront important. Dans quelque domaine de conception de MOSFETs (par exemple concentration de dopage ou épaisseur de couche d'oxyde) les couches sont tellement minces que nous pouvons déjà compter les atomes. Dans les autres domaines, ces effets quantiques vont entrer la scène dans le future proche. Pour aller plus loin que régime 10 nm, c'est nécessaire de modéliser ces effets. Mais les simulations précises prennent trop de temps. Facilités de calcul à haut performance créent une opportunité pour ces simulation d'être fais raisonnablement vite et avec la précision désirée.

Dans ce travail, l'objectif est de proposer et implémenter un algorithme de parallélisme pour un outil de simulation des effets balistiques dans les MOSFETs à l'échelle nanoscopique à doubles grilles. Cela est fait dans le but de diminuer le temps de simulation

pour arriver à utiliser les modèles plus précises et faire les recherches paramétriques nécessaire pour conceptions optimisées. Le composant considéré est étudié dans les aspects physiques et mathématiques. Outil de simulation considéré est étudié en détailles et les possibilités de parallélisation dans logiciel MATLAB sont considérés et comparés. À la fin, l'algorithme de parallélisation est implémenté et le code parallèle est généré et exécuté sur une facilité de calcul à haute performance de Calcul Canada. Les résultats sont présentés, validés (par comparaison avec les résultats de code original) et étudiés concernant l'efficacité de parallélisation.

Méthodologie

NanoMOS, un outil open source développé par les chercheurs à l'Université Purdue, est un simulateur de MOSFETs à double grille à l'échelle nanoscopique. Il considère transport balistique et résout les équations Schrödinger et Poisson pour trouver les distributions de charge et courent à l'intérieur de composant. Dans ce travail, le code de NanoMOS a été étudié concernant les possibilités de parallélisation et un algorithme de parallélisation a été implémenté sur ce code. Le résultat a été testé en utilisant les facilités HPC de Calcul Canada. Les résultats de simulation et les facteurs d'accélération de temps sont présentés et discutés pour différent nombre de nœuds de calcul. Ce travail, étant une expérience de parallélisation, peut être utilisée dans implémentation d'algorithmes parallèles sur les modèles plus complexes qui facilitent les études paramétriques et optimisations nécessaires pour un design précise de composants nanoélectroniques.

Dans l'exécution parallèle d'un code, la charge de travail est divisée et distribuée parmi différents nœuds de calcul. Idéalement la durée de l'exécution de code va être divisée par le nombre des processeurs engagés. Aussi la mémoire totale disponible sera le résultat de

multiplication de la mémoire d'un processeur avec le nombre des processeurs. Ce qui est important dans parallélisation, c'est la façon de diviser le job et la méthode de communiquer parmi les nœuds [15].

Calcul Canada est un collectif de consortia des infrastructures HPC, géographiquement distribués mais travaillant dans un système unique nationale. Un compte de Calcul Canada, offert aux professeurs et chercheurs, donne la possibilité de travailler avec les machines de calcul dans toutes les infrastructures de Calcul Canada et de profiter des supports techniques, programmes éducatifs etc. [21].

CLUMEQ, l'un des consortia de Calcul Canada, est composé de deux superordinateurs situé à l'Université McGill, Montréal et Université Laval, Québec [22]:

Colosse : une grappe de 960 nœuds (chacun avec 8 cores processeur et 24 GB de RAM), totalement 7680 cores et 23 TB mémoire, avec réseau infiniband QDR et capacité de système de fichier de 1 PB [22].

Krylov : une grappe de 48 nœuds (21 nœuds avec 4 cores et 8 GB de RAM et 27 nœuds de 8 cores et 16 GB de RAM), totalement 300 cores, avec réseau infiniband SDR [22].

De côté logiciel, NanoHUB est un ressource sur ligne éducationnel et académique dans la domaine de nanotechnologie. Il inclue cours sur ligne et documents d'apprentissage ainsi qu'un grand nombre de simulateurs de phénomènes et composants nanoscopiques. NanoHUB est formé par contributions de plus de 600 chercheurs et éducateurs et le nombre de ses utilisateurs augment rapidement [34].

NanoMOS est un simulateur open source offert par NanoHUB et écrit avec MATLAB. L'utilisateur a le choix de faire rouler l'outil NanoMOS à distance ou télécharger le code et l'exécuter sur son ordinateur [29], [34].

Modèle conventionnelle de transport de transporteurs est dérivé d'équation de transport de Boltzmann. Cette modèle est basée sur diffusion de transporteurs et décrit bien le cas où canal est longue. Mais dans les composants avec les canaux très courts, modèle quasi-balistique de transport doit être employé, comme il est employé dans NanoMOS. Dans cet outil, formalisme de Fonction non-équilibre de Green (NEGF), qui est une méthode de résoudre des équations différentiels inhomogènes, est employé pour résoudre l'équation de Schrödinger et calculer les densités de charge et courent. En plus, Équation de Poisson est résolu avec méthode de Newton pour calculer profil de potentiel [20], [29].

NanoMOS a été créé plus tôt avec un regarde algorithmique et mathématique, et pas avec un stress sur codage. C'est parce que le défi, ici, c'est de trouver des solutions pour les problèmes de modélisation physique et résolution des modèles complexe mathématiques et pas faire un codage optimal. MATLAB a été choisi comme la langue de programmation pour profiter de ses fonctions avancées. NanoMOS a environ 5000 lignes de code distribué dans différents fichiers fonctions de MATLAB qui sont appelés selon le type de simulation demandé par utilisateur [36].

Pour décrire brièvement la façon de fonctionnement de ce code, il faut dire qu'à part des étapes de donner les entrées, initialisation et calculs primaires, NanoMOS fait ses calculs en un nombre des itérations d'une boucle de calcul. Cette boucle doit satisfaire un paramètre de condition qui représente la convergence de réponses produits par les itérations. À l'intérieur de cette boucle, deux étapes sont pris : au début, fonction

chargenanomos.m est appelé pour calculer la densité de charge en utilisant le modèle de transport choisi. Après, fonction poisson.m est appelé pour calculer le nouveau profil de potentiel en utilisant les densités produites par chargnanomos.m. La différence maximum entre nouveau profil de potentiel et ancien profil de potentiel est comparée avec une critère de convergence, défini par l'utilisateur. Si le critère n'est pas respecté, la boucle est répétée [28].

Après avoir compris le fonctionnement du code, il faut l'étudier pour trouver les stratégies possibles de parallélisation. NanoMOS, en bref, performe un série d'algorithmes de solution des équations sur les nœuds d'une grille de l'espace. Cela est répété pour chaque point de bias. Dans quelque sous-programme de NanoMOS, ces opérations sont faites pour tous les « subbands » et « valleys » d'électrons aussi. Considérant cette structure, quelque stratégie de parallélisation est suggérée. La première c'est de distribuer les calculs en distribuant la grille de l'espace parmi les processeurs. L'autre est de paralléliser sur les points de bias. L'autre solution pouvait être parallélisation de l'algorithme de calcul (effectué sur chaque point de grille et bias) soi-même. Ça vaut dire trouver les possibilités de parallélisation de l'algorithme de solution des équations [28].

Parallélisation sur les points de bias est une solution simple et directe et en même temps inclusif et lourde en termes de routines. C'est simple et directe, parce que c'est l'algorithme le plus claire. Dans NanoMOS, la boucle de calcul principale (solution auto-cohérente) est répétée pour tous les points de bias. C'est inclusif et lourd parce qu'avec cette stratégie, presque toutes les routines de calcul sont engagées dans parallélisation. Chacun parmi eux doit être examiné pour compatibilité avec algorithme parallèle. Ses variables doivent être passés et retournés correctement. Finalement, quand un grand nombre de routines sont

inclus, au moment de faire des modifications ou ajouter des lignes de code, plus d'erreurs doivent être réglées. Les calculs reliés aux points de bias sont indépendants de l'un à l'autre. Elles ont seulement besoin de quelque initialisation, qui est généralement fait à l'extérieur de partie parallèle de code [28].

Pour implémenter l'algorithme de parallélisation sur les nœuds de grille, quelque décision doit être prise. Dans cette approche, des modifications sont réalisées à l'intérieur de toutes les routines qui travaillent avec nœuds de grille. Chaque routine reçoit ses entrées dans la forme des grands matrices qui incluant les données pour tous les nœuds. La routine fait ses manipulations et produit les matrices de sortie. Alors modifications parallèles sont faites indépendamment dans chaque routine. Cela augment « overhead » de cette stratégie. En plus, il faut voir si les routines ont des boucles assez longues pour chaque point de grille. Cela est nécessaire pour que la parallélisation soit efficient. Parce qu'avec les boucles courtes (peu de calcul pour chaque nœud) créer un nouveau nœud de calcul n'est pas profitable. En effet, la plus part des routines n'ont pas des boucles longues désirées. Une solution algorithmique peut être de diviser tous les nœuds de grille en quelque groupe et envoyer les calculs de chaque groupe à un processeur [28].

La possibilité de parallélisation sur « valleys » et « subband » est plus limitée que les deux solutions discutées. Seulement deux routines répètent leurs calculs pour les « valleys » et « subband » et ils sont, tous les deux, des parties de solution d'équation de transport. Alors dans chaque itération de solution auto-cohérente, juste la moitié d'algorithme est parallélisée. Une solution peut être de paralléliser sur les points de grille quand c'est mieux et paralléliser sur les « subbands » et « valleys » quand c'est préféré. De toute façons complexité et fragilité de cette solution est remarquable [28].

Parallélisation d'algorithme ne semble pas très prometteur non plus. La plus part des calculs sont effectués dans la boucle auto-cohérent, mais les deux routines principales de cette boucle (chargenanomos.m et poisson.m) sont dépendants de résultats de l'un à l'autre. Alors ils ne peuvent pas être exécutés en parallèle. Les autre possibilités de parallélisation d'algorithme ne sont pas assez significatif [28].

Pour conclure, quand il y a plus d'un point de bias, parallélisation sur les points de bias est très efficient; Parce que la plus part de charge de calcul est parallèlisée et aussi parce que peu de communication nécessaire. De l'autre côté plus de routines sont engagées et les paramètres et les appels des routines sont plus nombreux et l'algorithme est plus complexe. Quand il y a juste un point de bias, un mélange des autres solutions doit être utilisé. Dans ce cas, la solution n'est pas aussi efficiente. Parallélisation est moins complexe pour chaque routine, mais il y a un nombre considérable d'opérations parallèles à concevoir et réaliser. Ça vaut dire que les sessions parallèles sont indépendantes pour chaque routine et ça augment « overhead » considérablement [28].

Dans ce projet, NanoMOS est parallélisé sur les points de bias. Une série de points de bias avec tensions de grille identiques et tensions de drain variantes est utilisée pour les simulations. Fonction « parfor » de MATLAB est utilisée pour paralléliser la boucle auto-cohérent de code. Le code parallèle est exécuté sur une grappe et les résultats sont présentés [28], [39].

Codage parallèle dans MATLAB est fait en utilisant « MATLAB Parallel Computing Toolbox » qui inclut les fonctions parallèles et les fonctions de préparation de réseau de nœuds de calcul. « MATLAB Distributed Computing Server » est utilisé pour gérer la

grappe et communiquer avec « scheduler » ou « job manager » de grappe. Les nœuds de calcul sont appelés « worker » et le nœud principal est appelé « client » [39].

Résultats

Dans ce projet, la diminution de la durée d'exécution de code est le but principal. Facteur d'accélération est le paramètre qui représente augmentation de la vitesse d'exécution. Les limitations de facteur d'accélération sont discutées. En bref, la partie non-parallèlisable de code impose une limite au facteur d'accélération [20], [40].

Comme il est discuté en haut, parallélisation sur points de bias produit le moins « overhead » parmi toutes les solutions et abandonne très peu de la charge de calcul à l'extérieur de partie parallélisée de code. En d'autres termes, cette solution a le plus court route critique (route nécessairement sérial d'exécution) [20], [28].

Code parallèle est exécuté sur 2, 4, 8, 16 et 28 nœuds sur grappe Krylov de CLUMEQ. Points de bias sont générés avec tensions de drain qui augmentent de 0 V à 1.62 V avec les étapes de 0.1 V. Quand la tension dépasse 1.62 V (pour 28 nœuds) taille d'étapes est diminuée à 0.06 V pour éviter les tensions élevée de drain qui effectue la vitesse d'exécution de code. [22], [39].

Facteur d'accélération est présentée pour différents nombres de nœuds de calcul dans figure 3.

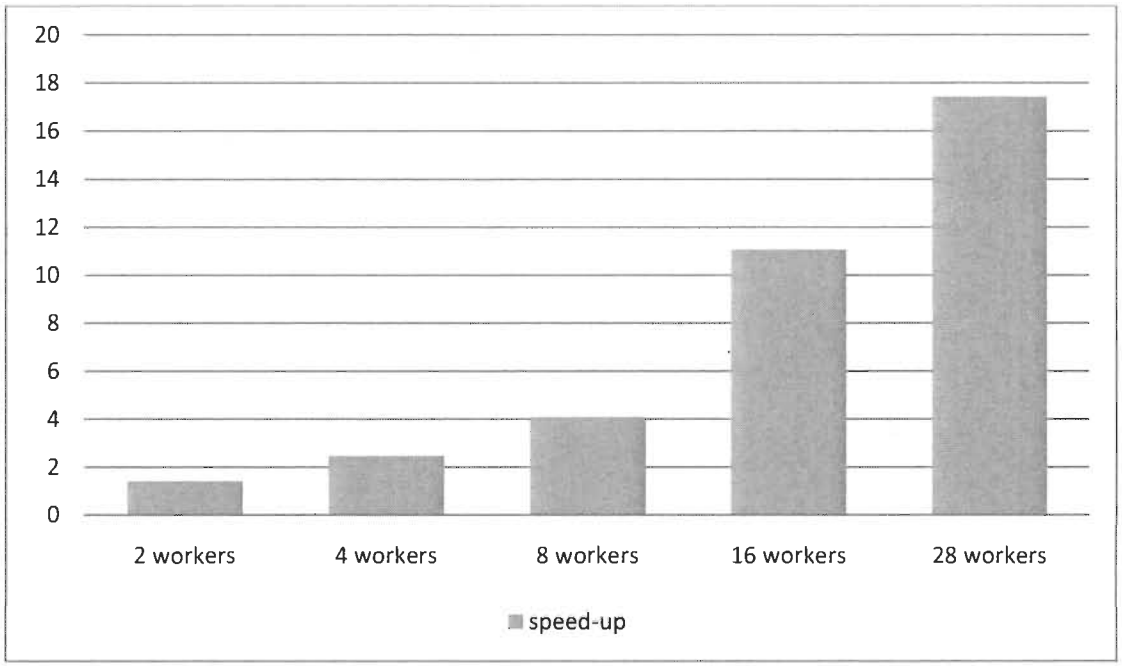


Figure 3 Facteur d'accélération mesuré par auteur pour différents nombres de processeurs

Considérant les discussions sur algorithme choisi, il est possible de dire que l'augmentation du facteur d'accélération ne sera pas saturée facilement avec une augmentation de nombre des nœuds et la tendance vue dans figure 3 peut être répétée pour les nombres plus grand de nœuds de calcul.

Validation quantitative de ces résultats est faite par comparaison de résultats de code parallèle et code originale. C'est notable que modifications reliées à parallélisation ne changent pas les fonctionnalités de calcul de code. Alors les résultats produits seront identiques. Le vecteur de convergence des donnés est utilisé pour cette validation.

Pour voir l'effet de nombre de « workers » sur efficacité de code parallèle, les facteurs d'accélération produits sont comparés avec la prédiction idéale (sans considérer les communications, initialisations et tolérances) dans figure 4 [28], [41].

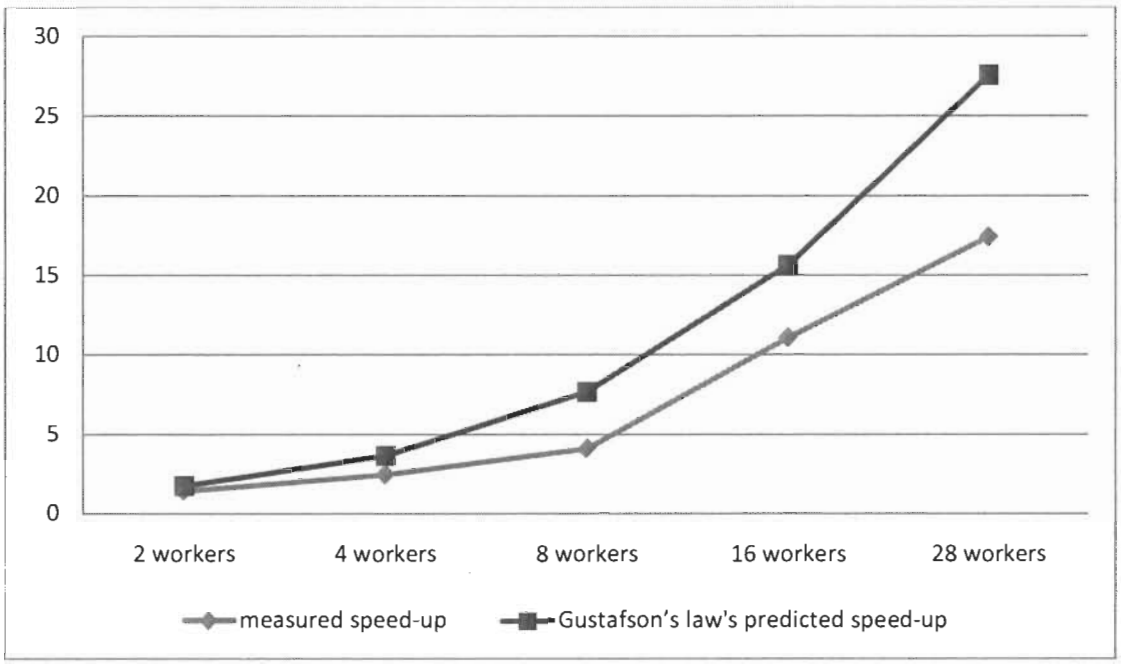


Figure 4 Facteur d'accélération mesuré par auteur, et facteur d'accélération théorique

Pour pouvoir mieux comparer, les résultats normalisés aux prédictions théoriques idéales sont présentés dans figure 5.

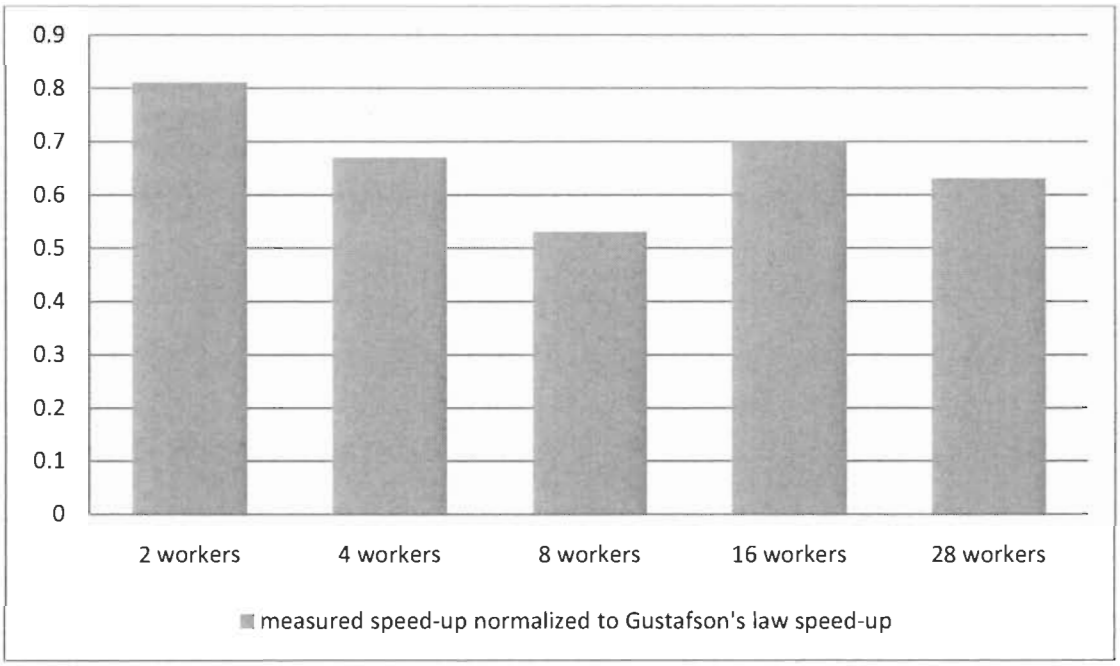


Figure 5 Facteur d'accélération mesuré par auteur normalisé pour prédiction de la loi de Gustafson

C'est difficile à reconnaître un motif dans figure 5. La raison est que la durée de l'exécution de code parallèle est de même ordre de grandeur que les tolérances de grappe utilisé. Il a été un des problèmes graves au moment de faire les mesures. Différents techniques ont été utilisés pour diminuer l'effet des tolérances. De toute façon, une pente vers le bas est visible quand l'ensemble des résultats sont considérés. Ça vaut dire que quand le nombre des nœuds de calcul augment, les mesures suivent la prédiction théorique moins étroitement. Cela est à cause de la petite augmentation de la durée de l'exécution pour les points de bias avec tensions de drain plus élevées. Cela cause le code parallèle souffre plus que le code série, alors efficacité diminue. Mais cela ne justifie pas l'utilisation de point de bias concentrer à proximité de zéro (seulement pour avoir des résultats plus idéal), parce que ça ne sera pas la façon que le code parallèle est destiné à être utilisé en

pratique; parce que le changement de point de bias est probablement nécessaire pour optimiser une conception.

Conclusion

Un algorithme de parallélisation est proposé et implémenté sur le code d'un outil de simulation balistique de MOSFETs à l'échelle nanoscopique à doubles grilles. Pour créer une base théorique, différents concepts des composants nanoélectroniques sont étudiés. le composant considéré est étudié dans les aspects physiques et mathématiques et technologiques et les obstacles de sa modélisation sont considérés. Calculs parallèles est proposé comme une méthode de conquérir les obstacles de simulations précises et complètes de ce type de composants. Facilités de calcul à haute performance sous le nom Calcul Canada sont présentés et étudiés comme infrastructures de cette recherche. Un outil de simulation de composant considéré est choisi et étudié en détaille. Les scénarios possible de parallélisation de cet outil dans logiciel MATLAB sont présentés et discutés. Le meilleur algorithme de parallélisation est choisi et implanté et un code parallèle est généré qui est exécuté sur infrastructures de Calcul Canada. Les résultats de calculs parallèles sont présentés et validés. Les résultats sont étudiés concernant l'efficacité de parallélisation. Facteurs d'accélération produits sont satisfaisants et la diminution de la durée de simulation est atteinte comme c'était attendu. Algorithme et code parallèle généré sont accessibles pour travaux additionnelles des chercheurs.

Chapitre 1 - Introduction

1.1 Nanoscale electro-thermal phenomena

Nanotechnology in definition is the science of manipulation of systems with at least one dimension in the range of 1 to 100 nm. Nanoscale engineering uses different properties of materials in this scale to create new devices [1].

Nanotransistors are one of the most important nanoscale systems and are now subject to great research and study and hopefully will replace today's transistors in near future. Naturally designing and producing nanotransistors with desired characteristics is of great importance. In the following discussion we see that temperature and thermal phenomena, with many other factors, have a huge effect on transistors characteristics. So it is essential to be able to model these effects correctly and efficiently.

Transistors, as one of the main nanosystems, are affected by temperature in different ways: First, transistor delay changes with temperature: higher temperature lowers the threshold voltage and decreases the mobility, causing the delay to change (increase or decrease depending on the strength of each effect).Second, high temperature increases the resistances, causing the interconnect delays to grow. Third, leakage power changes radically with temperature changes. The relation is in fact exponential. Leakage power itself can be very dangerous. This current increases the temperature, so a positive feedback is formed that can lead to the burning of the device. Fourth, reliability problems, for

example negative temperature bias instability (NBTI), oxide breakdown, and electro migration increase with high temperature too [2], [3].

In technologies where the isolation from silicon substrate is created using buried silicon-dioxide layers, because this isolator has a low thermal conduction, self-heating is a more serious problem. Examples are silicon-on-insulator (SOI), strained-silicon-on insulator (SSOI) and tri-gate CMOS transistors [4].

The scaling of transistors has continued for decades now, providing an increasing processing power. Today, this scaling process is facing one major roadblock: power problem. In current technologies, more scaling leads to power densities, heat generation and chip temperatures so high that the functionality and reliability of operation of the transistors are challenged. Today power density in chips is on the order of $100 W / cm^2$ [5].

In addition to increased power densities, non-uniform heat generation, which appears in nanotransistors, creates sharp temperature gradients in these structures [4].

High temperature can cause a wide range of problems: electrostatic discharge (ESD) causing malfunctioning, electro migration and electrical overstress, leakage current, timing failure, self-heating, hot spots, thermal stresses and so on[4].

Classical laws of physics do not predict the behavior of systems in nanoscale correctly and quantum physics must be employed to predict heat transfer in these scales. For example, Fourier law cannot model the thermal conductivity of super lattices and Boltzmann law is not successful in predicting radiation across small gaps[6].

Some of the properties of heat transfer in macro scale are not present in nanoscale, some change. In nanoscale heat conduction is ballistic and similar to thermal radiation. On the other hand thermal conductivity becomes dependent of material property [6].

In very small scales a phenomena called nanometer-scale hot spot occurs in drain region of a transistor, causing an increase in the drain series resistance and source injection electrical resistances. In non-traditional transistor geometries using novel materials, for example in ultra-thin body, FinFET, or nanowire transistors, this effect is stronger and heat conduction is more affected. Also increased surface to volume ratio in new geometry transistors increases the effect of material boundary thermal resistance in the modeling of transistors that needs attention [5].

New geometries make it more difficult to remove the heat generated inside the device and new materials used in new devices have much smaller thermal conductivities than bulk silicon (Table 1-1). Also when one or more of device dimensions are less smaller than electron or phonon mean free path, sub-continuum effects, like ballistic electron and phonon transport, create hot spots in drain and change the thermal modeling of device. In addition to that, the non-local transport conditions displace these hot spots in a different way than classical heat diffusion theory prediction and produce a higher temperature rise than what is predicted by that theory. Just to make it worse, when a transistor has semiconductor films that are thinner than the phonon mean free path, the thermal conductivity of these films is reduced because of phonon confinement and boundary scattering [5].

Table 1-1 Thermal conductivities of a few materials used in semiconductor device fabrication, phonon boundary scattering significantly reduces the thermal conductivity of a 10 nm thin silicon film

Pop, E., Goodson, K.E., "Thermal phenomena in nanoscale transistors", *The Ninth Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems, ITHERM '04.* 1-4 June 2004.

Material	Thermal Conductivity (W/mK)
Si	148
Ge	60
Silicide	40
Si (10 nm)	13
SiO_2	1.4

Electron transport is called electrical current and phonon transport is heat flow. When these two particles' transport is done under Ballistic conditions at nanometer length scales, a strong non-equilibrium between these energy carriers is caused. It means that the electron-phonon interaction will no longer be energetically or spatially uniform. Electrons transfer their energy to the lattice by interacting with phonons. Optical phonons have no role in the thermal conductivity. It is longitudinal acoustic phonons that transport heat. This is the process of generation and transportation of heat in a transistor: applied voltage creates an electric field with its peak happening in drain. The electric field accelerates and gives energy to free charge carriers which, in the case of a MOSFET are conduction band or free electrons. Heated electrons scatter with many particles: each other, lattice vibrations (phonons), interfaces, imperfections or impurity atoms. But only their scattering with

phonons can transfer energy. In silicon and most of other semiconductors, optical phonon emission is responsible for most of high field Joule heating. So the phonons take energy from electrons and the lattice is heated by Joule heating. When the lattice's temperature increases, electronic transport properties of the environment change. For example, in bulk silicon the electron mobility decreases as $T^{-2.4}$ in room temperature [5].

Optical phonons which take energy from electrons are slow moving particles that do not have any role in heat transport. But they decay into acoustic phonons which move fast and transport energy. Electron-phonon scattering time is in the order of tenths of picoseconds. But optical to acoustic phonon decay time is on the order of picoseconds. So that creates a population of optical phonons waiting to decay to acoustic phonons. But now if the generation rate of optical phonons from electrons in the process of Joule heating is greater than the rate of decay into acoustic modes, the population of optical mode phonons increases which, as mentioned above, affects electron transport properties of the environment. In the micro-transistors, the volumetric Joule heating rate is the dot product of the electric field ($\vec{E}$) and current density ($\vec{J}$). But in nanotransistor this is not the case. Electrons are accelerated by electric field mostly in drain, where the field is strongest. Then electrons travel a "mean free path" before scattering with phonons and giving their energy to the lattice. Electrons release their energy to the lattice in several steps over several mean free path travelling. That creates a non-locality of phonon emission from the peak of electrical field position in nanotransistors. The electron-phonon mean free path is calculated as the product of electron-phonon scattering time, which is almost 0.05 Pico seconds in the high electrical field, and electron saturation velocity (10^7 cm/s in silicon). So each mean free path is around5 nm. In micro-scale transistors this distance is negligible compared to

distances around some microns, or a little smaller. But in nanoscale transistors with channels with the lengths near 10 nm, it is a big displacement of energy. In other words nanometer-sized hot spots are displaced as much as some nanometers from the prediction of continuum theory about their position in drain. This in very small transistors is a significant change and must be modeled to predict the heat transportation correctly [5], [7].

By scaling, the density of devices in a chip increases, but at the same time there is almost no decrease in dynamic power dissipation, so volumetric heat generation density increases in a rate equal to scaling rate. When the dynamic power of due to nanotransistor technologies is considered: $C_{eff} V_{dd} 2f$ it seems that reducing operating voltage (V_{dd}) will be ideal for reduction of dynamic power. But to keep the drive current high enough, reducing V_{dd} must be accompanied with a reduction in threshold voltage (V_t). But source-drain leakage grows exponentially when V_t is decreased. So it is obvious that there is a tradeoff between performance and power consumption. When threshold voltage is lowered too much to lower the power, the performance suffers. So during the history of scaling, there has been a strong tendency to keep V_{dd} and V_t high and they have been scaled very lowly compared with the physical dimensions of devices. So over time transistors have had stronger electric fields and higher power densities. Now the fundamental barriers ahead of this technology have stopped it from continuing to go forward in expense of field magnitude and power density and a major shift in process design, performance targets, and architecture seems inevitable [8].

Dynamic power, being related to impedance, has an inverse relation with temperature. But leakage power increases with junction temperature. It means that if the increase in

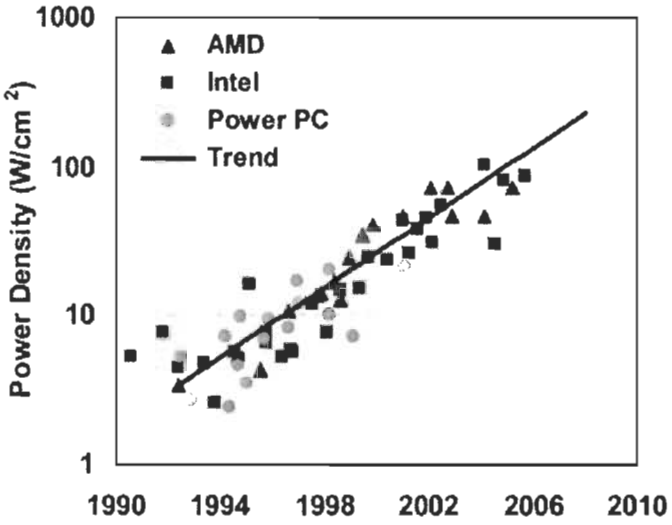
power or thermal resistance happens, as it is happening in new technologies, leakage power will grow too. So a positive feedback is formed in which leakage power and temperature start to grow [8].

To sum up so far, as device size reaches 10 nm, circuit density passes one billion devices per centimeter. There is an agreement that the technological roadblock of scaling trend at this point is the power problem. In this view, power densities, heat generation, and chip temperatures are approaching levels so high that reliable operation of devices is no longer guaranteed (Figure 1-1)[9].

Here a more detailed discussion of heat transfer mechanism in lattice is presented. Phonons are Eigen modes of vibration of atoms in the solid state of lattice. This phenomenon is used in describing thermal behavior of materials. These vibrations are divided in two categories of longitudinal and transverse modes. In a different categorization, depending on if the neighboring atoms oscillate in phase or out of phase the phonon associated with the vibration is an acoustic or an optical phonon respectively. The name optical phonon comes from the fact that an atomic dipole is formed by out of phase vibration, can interact with photons. High energy electrons generate optical phonons which in turn have to decay to acoustic phonons so that heat can be transported from the hot area of lattice. The ballistic travelling of particles makes the bulk parameter of thermal conductivity, which has been used to describe heat conduction in electronic thermal management models, an inaccurate approximation of heat transport for very short times, or very short distances [1].

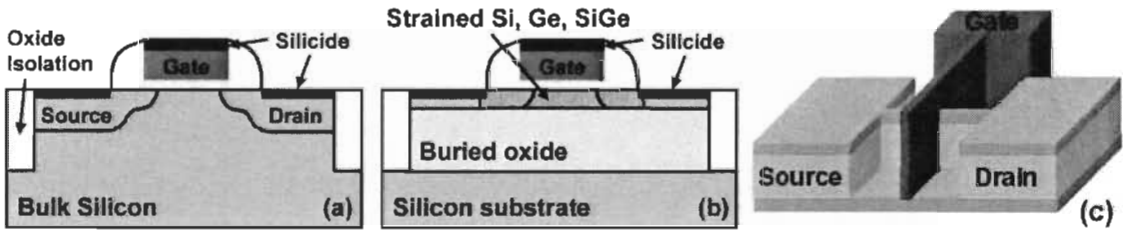
Although thermal conductivity and heat diffusion equation are still used to explain the majority of thermal transport phenomena at nanoscale, with increasingly high drive fields

they stop responding to the changes, because in these conditions different populations of electrons and phonons are not in equilibrium. In other words, the situation is so far from classical conditions that a single temperature cannot be measured for the lattice. This non-equilibrium of electrons and phonons and also the non-equilibrium between optical and acoustic phonons create nanoscale hot spots in short channel transistors. Another factor increasing this hot spot effect is the ballistic electron and phonon transport. Another phenomenon that changes the heat transfer in nanotransistors is thermal boundary resistance, formed at the interface of two solids. Currently there is not any theory to describe the thermal boundary resistance and thermal resistance of nanometers-thick layers. In some tries to describe these effects phonons are supposed to be particles that transport through an interface in a diffusion reflectance model. Sometimes phonons are supposed to be waves and their transport through an interface follows acoustic mismatch model. These theories have not been able to explain some of the experimental results. The choice between models depends on roughness of surfaces and scattering at the interface. Thermal conductivity is a bulk parameter; it is the average of effects of many different forms of phonons representing different modes of lattice vibrations. Because it is difficult to calculate each mode's exact share of heat transport, the complete understanding of heat transport in nanodevices is not yet reached [1].



Pop, E., Sinha, S., Goodson, K.E., "Heat Generation and Transport in Nanometer-Scale Transistors", *Proceedings of the IEEE*, vol. 94, Issue 8, pp. 1587 – 1601, Aug. 2006.

.Figure 1-1 Device power density growth over the time. Interesting to note is that the vertical axis is logarithmic and the horizontal one is linear so an exponential trend is obvious. Some fact to compare: the power density in a hot plate is in the order of 10 W/cm^2, in a nuclear reactor in the order of 100 W/cm^2 and at the surface of the sun about 7000 W/cm^2. Data compiled by F. Labonte, Stanford



Pop, E., Sinha, S., Goodson, K.E., "Heat Generation and Transport in Nanometer-Scale Transistors", *Proceedings of the IEEE*, vol. 94, Issue 8, pp. 1587 – 1601,Aug. 2006.

Figure 1-2 (a) bulk silicon (b) (strained) silicon/germanium on insulator (c) multiple-gate or FinFET devices. In more advanced geometries (b,c)

thermal conductivity of materials is lower and heat removal is harder. In b and c for the gate length of 10-nm, the thickness of semiconductor is 30% to 50% of the gate length. (c) is from the ITRS [7]

In advanced geometries heat removal is more complicated and harder and new materials, necessary to develop these architectures have lower thermal conductivities than bulk silicon (Table 1-1). It was said that self-heating in nanodevices is due to interaction of electrons and phonons. We saw how electron's mean free path in bulk silicon is about 5–10 nm. Phonons also have their own mean free path that can be calculated to be about 200–300 nm in bulk silicon at room temperature. When channel lengths in future technologies reaches both these limits, ballistic transport will be the main transport mode of both electron and phonon, in other words, both current and heat. These ballistic transports cause the energy carriers to have a non-equilibrium distribution. It was explained how different phonon modes affect heat flow differently. Worse yet, because of phonon confinement and boundary scattering in geometries where semiconductor film is thinner than phonon mean free path, the thermal conductivity of the environment will be dramatically less than bulk silicon. So in thin-film and confined-geometry transistors [Figure 1-2 (b) and (c)] heat flow and cooling of transistor is more difficult and temperature is higher than bulk transistors [Figure 1-2(a)] with equal power input [9].

Joule heating, dot product of the *macroscopic* electric field and current density fails to approximate the *microscopic* non-locality of phonons in drain. This average model gives no understanding of electron energy exchanges with various phonon modes and it does not provide a spectral understanding of phonons either [7].

Electrons lose their energy by electron-phonon scattering at a rate equal to the macroscopic relation: P=I.V. In channel lengths bellow 50 nm, phonon distribution

function affects the heat conductivity of the transistor. So the distribution of phonons must be taken into account in the design. In nanotransistors, for a drain size of around 10 nm where the field is strong, phonons receive energy from electrons in a distance up to 50 nm away from drain. Acoustic phonons transport the heat to the contacts and package boundaries. The propagation of acoustic phonon is faced a resistance when they scatter with other phonons, impurities, and boundaries. That causes a heat accumulation inside the device and creates higher temperature. More exactly, phonon population is increased inside the device and its effect is represented by a higher effective temperature [8].

Phonon boundary scattering can reduce the effective thermal conductivities of ultra-thin silicon films up to 10 times compared to that of bulk silicon [8].

Within some 10nm space around heat source, charge and heat energy transports do not follow the diffusion model. In this distance, electrons and phonons are in near-ballistic conditions causing highly non-equilibrium distributions of both electrons and phonons. If some certain quantum effects are small enough to be neglected, the solving Boltzmann transport equation (BTE) gives almost accurate distribution profiles for different phonon modes, leading to an almost exact profile of heat distribution [8].

According to simulations, in electric field strengths equal to that of modern transistors, almost 2/3 of the heat generation is done through generation of optical phonons. But they are responsible for little to no share of heat transport, because they are much slower than long wavelength acoustic modes. So it is absolutely necessary to find out how these optical phonons decay into acoustic modes [8].

To understand this decay it is necessary to know the optical phonon lifetime for different conditions, including different power densities [8].

This lifetime has been shown to be about 4 picoseconds for equilibrium conditions in 300 degree of Kelvin. This time decreases as power density of the channel increases [8].

As the channel length is scaled the power density increases. The relation between power density and channel length for transistors with channel length around 10 nm is expected to be: $L^{-1.7}$ [10].

The thermal conductivity of silicon thin films and nanowires in the transistors with channel lengths bellow 100 nm has been shown by experiment to be two to five times bigger than that of bulk silicon. This is caused by phonon surface and interface scattering [1].

In macro scale, many different carriers are involved in heat transfer: phonons, electrons, photons are molecules. They all behave like particles in these scales. In nanoscale on the other hand, carriers show wave behaviors a lot more often, so that the wave effects are often more important than particles behaviors, though that is not always the case[11].

Mean free path of a particle is the distance it travels between two collisions with other particles. These collisions can happen with two different types of particles. Nanoscale heat transfer phenomena can be seen in five categories depending on the relative sizes of mean free path and channel length [11].

When the structure is much bigger than the mean free path of the carriers, the diffusion regime can safely be used. This model then can be solved using conventional methods. Now depending on the relative size of the structure to the mean free path of energy carriers, i.e. electrons and phonons, two distinct regions can be identified. Inside each region a regime can be used that accounts for the wave properties of that carrier, instead of its

particle properties. There are also two transition zones where the size is comparable with the mean free path of a carrier and using one regime means accepting some approximation. In these zones scattering process that takes place in the environment determines if the diffusion regime should be used or ballistic regime. If scattering destroys the phase of the wave, the wave effects die out and diffusion can be used with some approximation. Inelastic scattering occurs when inelastic collisions, for example collisions between electrons and phonons or between phonons and phonons, destroy the phase of the wave. The when the structure is smaller than mean free path and the ballistic regime is used, it means that the scattering is neglected [11].

The ballistic behavior happens in highly scaled transistors when power density is above 1000 W/cm^3 and channel length is well below 100 nm, in other word much less than relaxation length. In this situation the behavior of "hot electrons" and "hot phonons" needs to be analyzed [12].

The non-equilibrium situation of energy transport in nanotransistors changes their behavior to a great extent. This phenomenon reduces the trans-conductance of the transistors, in other word, the drive current. Also it can increase sub threshold leakage power. These changes occur due to the fact that when electrons scatter and transfer their energy to optical phonons, because these phonon modes have low group velocities (~ 1000 m/s), they cannot transport the heat from the hot region. In 10 picoseconds they decay into fast moving acoustic phonons. But the delay means that there is an increase in energy density near hot spots, which affects electron transport [12].

So, nanoscale sizes create nanoscale hot spots in the transistor's drain, and that increases the drain series and source injection electrical resistances through creation of high

densities of carriers. This process is helped by use of new materials and modern geometries like ultrathin body, FinFET, or nanowire devices. Thermal analysis in these cases differs from classical diffusion theory and sub-continuum phenomena like ballistic electron transport play an important role in heat generation and heat transport [9].

Also in new geometries there is a greater surface to volume ratio which means boundary thermal resistance has a larger effect on heat transport in modern transistors [9].

Now that it is observed that heat conduction in nanostructures is much more limited than anticipated by the Fourier theory and that size effects produce higher temperatures, more effective thermal management measures have to be taken. As a positive side, these size effects might also offer the possibility of developing highly efficient thermoelectric (TE) materials and more efficient thermal management through direct cooling [13].

Size effects can cause limited heat conduction capability in super lattices of nanotransistors, which produce some thermal problems. But these effects also enhance the properties of thermoelectric (TE) devices, used in thermal management of semiconductor lasers. Maybe they have the same effect on thermal management of microelectronic devices too. In fact a factor called TE figure of merit (FOM) that indicates the efficiency of TE coolers, has shown high numbers in some III–V super lattices and TE super lattices meaning low thermal conductivities [13].

Heat transfer at macro scale follows some principle rules. One of them is the Fourier law that is basically a diffusion equation. Another useful factor to describe thermal behavior in macro scale is thermal conductivity which is a material property and independent of size and shape [6].

High power density and temperature in electronic circuits reduces transistor carrier mobility and threshold voltage and increases interconnect resistance, it produces reliability problems like electro migration, dielectric breakdown, and negative body biasing. It also increases power consumption by increasing sub-threshold current. At the end cooling costs increase because of all the negative impacts of scaling on thermal management [14].

To solve a heat generation problem within the classical limits, Joule heat generation rates are calculated as dot product of electric field and current density. Temperature profile can be found by solving heat diffusion equation derived from Fourier law. This method is applicable when the assumption that electrons and phonons, as current and heat carriers, are in local equilibrium is valid and also when the discussed sizes are bigger than some mean-free-paths of carriers [12].

Self-heating in semiconductor device is a production of scattering of electrons with phonons. To find a fairly exact model of this process all of the scattering events have to be considered in the simulation. Solution of the Boltzmann Transport Equation (BTE) for both electrons and phonons can provide this goal. BTE is a semi-classical transport regime in which charge and energy carriers are particles between scattering events [12].

1.2 Problem

In thermal aspect of micro and nanoelectronics two problems can be identified. One is the process of heat generation, heat transport and heat conduction in each device and the other one is the heat management at a system level [6].

Here we will discuss thermal analysis within each transistor.

Today, thermal management has to be applied at all levels of system. It starts from transistor and continues to the circuit and microarchitecture and finally to the packaging. However in modern thin-body devices with higher field levels thermal management within the transistors has become more important than before [10].

Before, hotspots used to appear only at the circuit level, due to the architecture of devices on the chip. Now, nanoscale hot spots are starting to appear inside the transistors [9].

Micro- and nanoscale electronic devices are capable of producing heat fluxes in small hot spots in the order of some thousands of watts per centimeter square. In the chip level some areas as big as a couple of hundred micrometers in diameter with a temperature $10-40^{\circ}$ C hotter than the surroundings can be formed because of difference in the work load of transistors. Naturally cooling systems are designed to fight the highest temperature on the chip and because the cooling systems usually cool all the area of the chip, this means that some of the cooling resources go to waste because of hot spots. If the hot spots are removed by correcting the design, cooling costs decrease, or there will be more cooling resources and more densities of devices become practical. The same goes for the nanoscale hot spots formed in the nanotransistors. In bulk silicon devices, the thick silicon substrate (some several hundred micrometers thick), with its high thermal conductivity, absorbs the heat from hot spots and distributes it to a size comparable to its own thickness. This is still much smaller than the whole circuit size, but one can observe that in this way bulk silicon substrate softens the extremely non-uniform power dissipation patterns. Now when the silicon wafer becomes thinner and when stacked chips or three dimensional chips are further developed, there will be much less heat dissipation from silicon substrate and hot

spots will be smaller and heat distribution function will be more steep, all that giving rise to higher temperatures [1].

Here heat flow paths will be discussed in a chip. Up to 70% of heat generated in a microprocessor exits through thermal interface materials and heat sink. 30% exits through multi-level interconnect dielectric structure, solder bumps and printed circuit board. To reach to a reliable system-level thermal analysis, both of these heat flow paths must be simulated. Interconnects are the part that have received little attention as heat flow path, but they have more potential for further studies. Also current thermal analysis is based on average power dissipation. But considering the growing non-uniformities in devices this has to change in future [4].

Investigations on nanoscale electronic properties started in the 1960s, but nanoscale thermal property is a new field of research. One reason might be that measuring thermal data in small scale is more difficult. In fact some new measurement techniques for temperature and heat flow at atomic scales have been developed in recent years: thermocouples placed at the tip of atomic force microscopes or scanning thermal microscopy (SThM) can measure temperature with 100-nm resolution. Sharp heated metallic tips in ultrahigh vacuum environment make scanning seebeck voltage measurement technique capable of 2–3 nm resolution measurements. Femto second laser technology measures heat transfer and acoustic velocities in thin-film sub micrometer structures. Thermo reflectance imaging in visible wavelengths measures the surface temperatures with some 100 nm spatial resolution and 1–100 mK temperature resolution [1].

Recently, as thermal aspect of design has become more important, researches have started to see interconnect heat dissipation and especially self-heating in modern transistors as an area with essential importance in device and circuit level design [8].

Although transport simulations have grown successfully recently, nanoscale self-consistent electro-thermal simulations are still rare and undeveloped [8].

In the near future the dominant transistor geometries will be ultra-thin body devices such as multi-gate MOSFET (FinFET) and silicon-on-insulator (SOI) [14].

Considering the complexity of these modern devices, being able to model thermal characteristics of a given IC quickly and efficiently, so that thermal effects of any change in design can be simulated before the design is complete, is becoming more and more complicated [14].

Presently the high computation demand of nanoscale electro thermal analysis makes it very difficult to alter designs in response to thermal concerns. To reach the efficient design of circuits based on nanotransistors, this obstacle must be overcome [15].

1.3 Role of parallel computations

Conventional thermal analysis techniques of chips are so costly in terms of computation resources and time that thermal analysis had no place in original circuit design. In fact, cooling was designed for the circuit at hand and some thermal optimization was performed only in the packaging and cooling system design. So the main opportunity to optimize the thermal characteristic, which is during circuit design, was lost. Now, fast thermal analysis techniques, based on classical Fourier transport, are emerging and they are employed to model heat transfer in chips and also cooling packages [14].

These techniques, being fast enough, still use Fourier heat flow model which is a classical model. So they are incapable of simulating devices in nanoscale, because they do not describe phonon quantum thermal effects correctly. However some techniques have been developed to model nanoscale phonon heat transport in device-level. Some of them are molecular dynamics methods, Boltzmann transport equation (BTE), and ballistic-diffusion model. Still computational complexity is a big obstacle in applying these methods in large-scale thermal analysis. Among them molecular dynamics methods are the most accurate models, because they directly simulate inter atomic interactions. Being so accurate, they demand extreme amounts of computational resources. The BTE methods simulate phonon transport and approximate ballistic phonon transport quite accurately. They are much less costly than molecular dynamics methods in terms of computational resources and it is much more practical to use them, but they are still very complex and are used only in device-level analysis. The ballistic-diffusion model is an approximation of the Boltzmann transport equation. It is more efficient than the two other models, but still its complexity and demand of computational resources are not comparable with the Fourier model. So there is a large distance between nanoscale chip-package thermal analysis models optimized for efficiency and those optimized for accuracy. To enjoy the benefits of both in a thermal analysis model that is applicable to IC design processes, this distance has to be shortened greatly [14].

During the last decade, several research studies of nanosystems have been done. But nanoscale heat transfer model is comparably a very young research subject [15].

One solution to the problem of simulating computationally demanding thermal analysis is employing parallel algorithms and implementing them on high performance computing systems [15].

1.4 Objective

As it was explained above, there is a need to create effective modeling tools for nanodevices. These tools include physical, mathematical and informatics aspects. The objective in this work is to contribute to creation and improvement of such modeling tools.

Modeling heat generation and heat flow problems in nanotransistors is closely related to electrical modeling of these devices. Many of the presently available modeling tools created by researchers in this field are more often at the stage of electrical modeling using quantum mechanics phenomena, rather than working directly on heat generation and heat flow. The later requires access to established electrical models, which is still an objective in this field of research. Therefore, working on a modeling tool that employs quantum effects to model nanoMOSFETs is a suiting objective for this thesis. Electrical modeling of nanoMOSFETs is the primary step in designing, optimizing and finally manufacturing next generation of MOSFETs. Modeling techniques that incorporate quantum phenomena to model nanoMOSFETs and generate carrier (electron and phonon) distribution data along the way pave the way for exact heat analysis too.

Another problem to face in this field, as explained in 1.3, is the fact that physical models and mathematical tools that are used to describe nanodevices are of such complexity that simulations that are based on them are too computationally expensive. To attack this problem parallel computation has been suggested as an effective solution. In

fact, employing parallel computation to facilitate simulations that perform exact modeling of nanoMOSFETs is a new approach to improve modeling tools in this field.

Considering above discussions, the objective of this thesis can be summarized as follows. A general understanding of nanodevices (with emphasis on nanoMOSFETs) and obstacles of their development is provided. A specific device is chosen and a modeling technique developed for that device is studied from literature in details. High performance computing algorithms and infrastructures are studied as tools of this research. A simulation algorithm that implements the studied modeling technique is taken as a base and different parallelization algorithms are considered to be implemented on it. High performance computation tools are used to generate and run a parallel version of the simulation algorithm in order to improve its results in terms of computation time.

1.5 Methodology

Following the objectives mentioned above, the methodology of the research can be described under three categories: gathering theoretical information and related literature, accessing the tools, realization of objectives and generation of results.

The first step, naturally, is literature review and providing the basic knowledge needed to go on with the project. This step includes study of electro-thermal phenomena in nanoscale, defining the main obstacles to be resolved in this field, an introduction to nanodevices, MOSFET operation principals, MOSFET geometry and fabrication processes, study of Double gate MOSFET and introduction to high performance computing. The main means of acquisition of this information is reading academic articles, books and theses and other scientific and technical documents. The usual method of finding these materials is

searching in IEEE archive of academic articles using IEEE official website or finding other academic articles using other online search engines, University library (mainly for books) and some specialized websites (ITRS, NanoHUB, Compute Canada...) for specific documents.

Finding and accessing the tools needed for this research refers to tools related to high performance computing and nanoMOSFET simulation tool and corresponding modeling technique. High performance computing infrastructures are gathered under Compute Canada platform. To gather information about the consortia and each machine and infrastructure and also to gain the permission to access these systems a researcher should open an account in Compute Canada website and seek remote access to the machines according to the regulations. By doing so, one can also benefit from support of technical staff who have specialty in operation of the machines and also in use of high performance computing in different branches of science and engineering. To choose a simulation tool and the corresponding modeling technique, nanoHUB, a cyberinfrastructure of nanotechnology related resources, has been used. NanoHUB provides educational documents and applications and also simulation tools in this field. Researchers in Canada can seek access to these resources through nanohub website. A simulation tool for double gate nanoMOSFETs (NanoMOS) was chosen as the simulation algorithm to base this work on. The modeling method used in this tool is explained in details in the PhD thesis related to this tool.

Realization of the objective of the research and generation of results includes study of selected simulation algorithm (NanoMOS) which is closely related to the modeling technique described in accompanying thesis, study of parallel algorithms applicable to the

simulation algorithm, study of options of high performance computing available and suitable for this project, applying the chosen parallel algorithm on the code of NanoMOS and finally running the parallel code on a parallel processing machine.

NanoMOS is an open source code written in MATLAB. It applies Newton method to Schrödinger and Poisson equations. These equations are derived from the physical modeling of the double gate nanoMOSFET. A parallel algorithm based on parallelization over bias points of MOSFET is chosen after discussing different options. This algorithm is applied to the code using MATLAB Parallel Computing Toolbox. Three different consortia of Compute Canada (CLUMEQ, sharcnet and Westgrid) are studied as candidates for running the parallel code. Also running the code on a parallel array of machines in computer Lab of Department of Electrical and Computer Engineering of UQTR is studied. MATLAB uses its Distributed Computing Server to form a parallel array of individual machines. Finally the code is run on Krylov infrastructure from CLUMEQ. The parallel code is run in different tests with different numbers of parallel threads and duration of each run is measured. The results are compared to the duration of execution of the serial code doing the same tasks, which represents the contribution of this work. The results of parallel code are validated by comparison to the results of the original serial code and some parametric analysis of the duration times are presented.

1.6 Thesis structure

This text is structured in seven chapters. In the first chapter (introduction), studied phenomena, the subject and goal of this study and also the main tools employed in this research are introduced. In second chapter different categories of nanoelectronic devices are introduced. Then the type of devices studied in this work, as well as their electro-thermal

constraints is discussed. In third chapter, High performance computing concept and infrastructures are introduced. Physical and mathematical models and concepts related to the chosen device, its fabrication and packaging methods and geometrical aspects and finally numerical approach of modeling are discussed in chapter four. In chapter five, mathematical formulation of the models used in simulation is discussed. Then algorithm of simulation is studied in detail and different possibilities of its parallelization are discussed. Then implementation of parallelization algorithm is explained. In chapter six, performance of parallel code is compared to the performance of serial code and also parallel code is validated by comparison of results. Chapter seven is a brief conclusion of this work.

Chapitre 2 - Nanoelectronics devices and Nano thermal constrains

2.1 Nanoelectronics devices

2.1.1 Introduction

Nanotechnology and Nano-science takes different names when it employs different phenomena and is applied to domains of electrical, mechanical or optic engineering: Nano Electro Mechanical Systems (NEMS), Nano Opto Electro Mechanical Systems (NOEMS). Among more detailed applications of nano in electrical engineering, the most important ones are:nanoelectronics devices and in particular nanotransistors, nanotubes and Nanowires [15].

NEMS and NOEMS: Since development of photolithography Technique, fabrication of micro-electronics devices has advanced a lot. Advances in micro-electronics and IC fabrication have had a great impact on development of micromechanical devices, Micro Electro Mechanical Systems (MEMS) and Micro Opto Electro Mechanical Systems (MOEMS) too. These techniques are the means of mass production and low prices, and at the same time they are high precision techniques, a feature that is essential to the sensitive nature of their products. Also miniaturization technologies of MEMS, in their advanced forms, help create NEMS and NOEMS [15].

NOEMS and NEMS are nanoscale analogous technologies of MOEMS and MEMS. NEMS are simply electro mechanical structures (sensors or actuators)that have at list one dimension with the size below 100 nm. Despite analogy, physics of NEMS and NOEMS is different from that of MEMS and MOEMS, because in nanoscale surface forces are even stronger and sometimes quantum effects describe the behavior of the system more accurately [15].

There are two main approaches toward fabrication of NEMS: Top-down and Bottom-up. In Top-down approach bulk material is taken and by different techniques is formed into a structure. Nano-Lithography and Nano-Printing are two technologies included in this category [15].

In Bottom-up approach individual atoms and molecules are manipulated and used to assemble molecules, particles and structures that join together and form a certain system. Auto-assembly and dip-pen lithography are two technologies included in this category. Dip-pen lithography is a method of manipulation of single atoms using the tip of an atomic force microscope [15].

Carbon nanotubes: Carbon nanotube is a very recent discovery. It was first seen and discovered using an electronic microscope in 1991. They have many astonishing properties. Carbon nanotubes are chemically very stable. Each carbon atom is in bond with three other and the whole structure is the form of a cylinder with a circumference of some nanometers and a length possibly as big as some micrometers [15].

Having high heat conductivity, carbon nanotubes can have some applications in electronics. Also they can play the role of heat removal element in combinations with

plastics. This will take away the most serious barrier of using plastics in engines and other applications where a lot of heat is generated [15].

Possible applications of nanotubes in electronics too seem very exciting. Chirality is a characteristic of each carbon nanotube which in brief is the angle in which the graphite sheet is rolled over itself to make the nanotube. Chirality determines if the carbon nanotube is a conductor or it is a semiconductor. By having this property, carbon nanotube are completely unique among all the materials. Also some types of them become superconductors in low temperatures. By connecting two metal electrodes by a nanotube, it is possible to make a Field Effect Transistors (FET). Electrons pass from the inside of the nanotube and it plays the role of silicon in conventional transistors. Additionally, carbon nanotubes waste very little electricity. In theory switches made from carbon nanotube can have 1000 times' higher frequency than conventional switches. If the obstacle of manipulation and assembly is overcome, they will definitely find vast applications in different NEMS systems like nanoengines, ultra high frequency oscillators, actuators, resonators, etc.[15].

Some methods have been suggested telling how to separate conductor and semiconductor carbon nanotubes, but more development is needed. Also when making a structure using nanotubes, the impurities that enter the structure along with nanotubes, have been a problem (catalyzer particles, amorphous carbon, carbon fullerenes). So some advances in purification are also needed for fabrication of nanotube structures [15].

Nanotransistors: For the past decades, scaling of transistors has made computers more and more powerful. But now fabrication limits and some quantum phenomena related to the

size and materials have slowed the scaling process of conventional field effect transistors (FET) [16].

To pass these barriers, some new devices, based on quantum effect and in atomic scales are being developed. These new devices that are supposed to replace conventional transistors and be used in ultra-dense circuits, have the ability to function as switches and amplifiers, as conventional transistors do. But they are based on different concept and are specially designed to perform successfully in nanoscale. FETswork with electrical currents flown by charge carriers that travel in masses through bulk semiconductor in the channel. nanodevices employ quantum mechanical phenomena that are more effective and even dominant in nanoscale. These charge carriers can move one by one through energy levels in lattices [16].

Here we will have a mostly qualitative discussion at length to explain how these next generation transistors differ from today transistors. There are two main categories of nanoelectronic devices:

1. Solid-state quantum-effect nanoelectronic devices
2. Molecular electronic devices [17].

Solid-state quantum-effect nanoelectronic devices are closer in structure and principles of operation to the conventional transistors. In these technologies fabrication methods that are being used will continue to exist and the geometries are more advance versions of conventional ones [17].

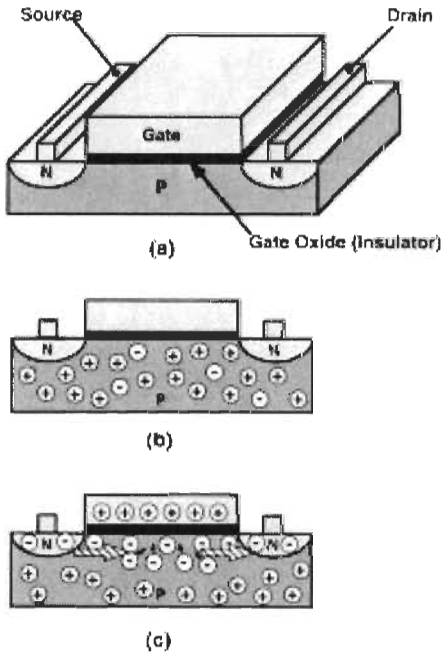
Molecular electronics instead employs completely different materials and structures than what exists now. This revolutionary change might have very beneficial features.

Molecules that are the structural basis of these technologies are already as small as a building block can be. Also all the molecules of the same kind are completely identical and the process of making them might be as simple and as cheap as mixing two solutions and they can be made in really huge numbers. All that is needed to be solved is what, if any, molecules and structures have the potential to form a controllable switch and also how to manipulate these tiny molecules and put them in their designed places in the circuits [16].

2.1.2 Structure and Operation of MOSFET

The metal-oxide semiconductor FET (MOSFET), with its low power consumption and efficient fabrication method has been the dominant transistor in the industry for decades [16].

As it was said above, novel designs for nanoscale devices employ different physical laws than in MOSFETs to let electrical current flow and to control it. But they all keep the basic concept used in MOSFETs. They all have a source, a drain and usually a gate. What changes most, in structures, is the material and geometry of channel [16].



GOLDHABER-GORDON D., MONTEMERLO M. S., LOVE J. C., OPITECK G. J., ELLENBOGEN J. C., "Overview of Nanoelectronic Devices", Proceedings of the IEEE, vol. 85 , Issue: 4, pp. 521-540, 1997.

Figure 2-1 Structure of an NMOS transistor

The expression "metal-oxide-semiconductor field effect transistor" contains the names of materials that construct the MOSFET. Doped silicon, in spite of pure silicon, which is a bad conductor, has semi-conductor-like characteristics. That is because dopants impurities create some mobile charges in the silicon crystal. Negatively doped or N-doped silicon receives free electrons and positively doped or P-doped silicon has some extra electron vacancies or holes. Both free electrons and free holes are charge carriers and can pass the current flow through silicon. MOSFETs have a crystalline substrate of doped silicon. Gate is a metal electrode that is insulated from semiconductor substrate by an insulating oxide. This is why it is called "Metal Oxide semiconductor". When the gate is charged to a certain voltage, the produced electric field in the substrate reaches a certain level than can affect

the movement of carriers and thus current flow in the semiconductor. For example in a p-doped MOSFET as in Figure 2-1, ordinarily even when there is a difference of potential between source and drain, the carriers are too few to carry electrical current. But when gate voltage is increased, electrons are attracted to the gate and the channel will have more positive carriers and current flows. The name "field effect transistor" comes from this process [2].

Miniaturization of FETs: A very common tendency is to develop enhanced versions of FETs for nanoscale needs. But a valid question is if any geometry built on the same concept can work in that scale. For example because such a chip has to have very narrow wires, current can flow between wires and performance is affected. Also the technology to fabricate such small structures, uniformly and efficiently, has yet to be developed [16].

Here it is discussed why the potential of scaled down FETs to play the role of nanotransistors is limited and why new generations of devices are expected to emerge:

1. When the FET is scaled to a great deal, the electric fields produced by bias voltage gets stronger and eventually it can cause an "avalanche breakdown" in which a high current suddenly flows through material and damages it [16].

2. Heating in FETs is inevitable. It even increases with scaling. So, increased densities necessarily mean increased heating in the unit of surface. Especially that heat conductance of transistor is limited. So scaling and density have limits and in fact right now, the technology has reaches some of these limits [16].

3. When the channel gets very small, the number of dopants becomes so small that the uniformity of their distribution in the channel is not valid anymore. So there will be two

solutions: one is to stop using dopants as in a GaAs hetero structure. The other one is to make an absolutely uniform doping using molecular nanoelectronics [16].

4. With extreme scaling the tunnel is so thin that electrons tunnel from source to the drain through the semiconductor, even when the device is off. On the other hand some nanoelectronic devices use the same phenomenon to control current [16].

5. The oxide insulator beneath gate prevents the electrons from tunneling between gate and drain. With scaling this leakage current grows too[16].

So in nanoscale, where the quantum physics effects become dominant, building a device that relies on bulk properties of materials and uniform doping is going to be an exceedingly difficult job. Instead, because energy quantization and tunneling are common properties of materials in that scale, quantum effect devices and molecular electronics that employ these properties are going to have obvious advantages [16].

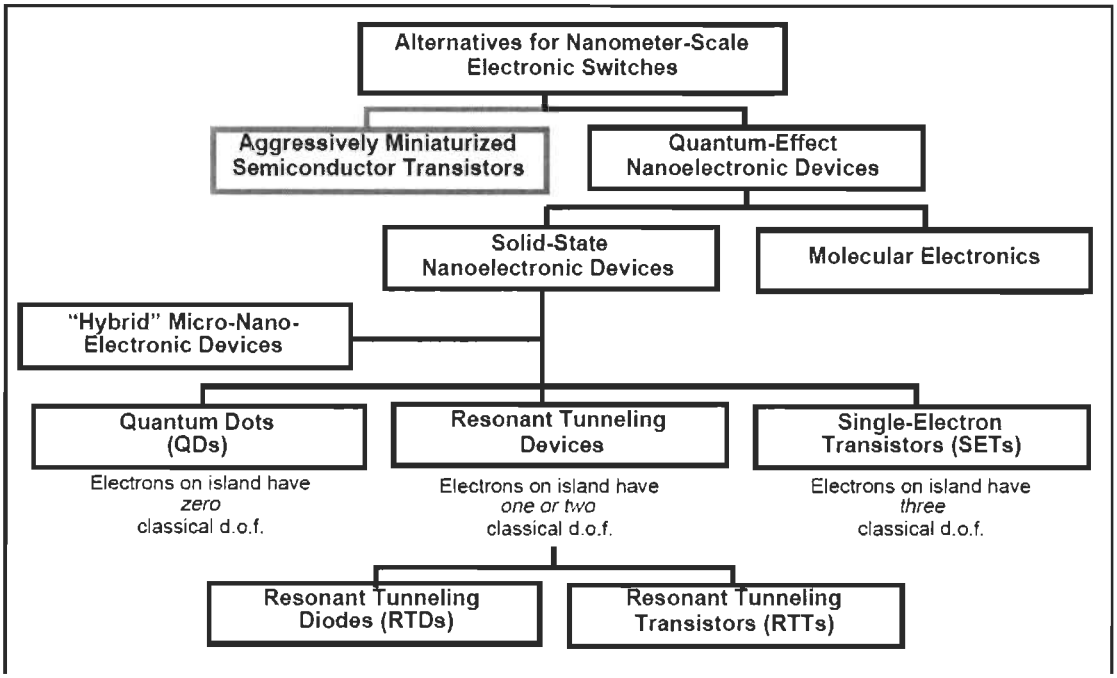
Solid state quantum effect devices and single electron nanoelectronic devices: These are some solid-state devices proposed to be used in nanoscale applications instead of the bulk-effect devices. All of these devices have a structural common element, called "island", which plays the role of channel in FETs. Island is a small region made of semiconductor or metal in which electrons may be trapped or confined. These devices employ quantum mechanics effects to control the displacement of electrons in and out of the island. Three categories can be identified in regard to the number of dimensions in which electrons are confined in the island [16]:

1. Quantum Dots (QDs) or artificial atoms: Electrons are confined in three dimensions and have zero degrees of freedom [16].

2. Resonant Tunneling Devices (RTDs): Electrons are confined in one or two dimensions and have two or one degrees of freedom [16].

3. Single-Electron Transistors (SETs): Electrons are confined in no dimensions and have three degrees of freedom. (Figure 2-2) [16].

By changing the shape and size and composition of an island, quantum effects act differently and island's properties and consequently device's properties change. As an example of composition, when an island is made of semiconductor, the electrons have a much more mobility in it than when it is made from metal. That is because the mean free path of electrons in metals is much smaller than semiconductors and passing electrons can travel through semiconductor with less collisions. So by changing the composition of the paths that electron might take, conductivity of the device can be controlled. For example adding a semiconductor path might let the current flow by increasing the conductivity. Also semiconductors made from elements from group III and group V of periodic table [gallium arsenide (GaAs), aluminum arsenide (AlAs),...] have higher motilities than silicon (group IV).It has been shown that we can make defect-free junctions between these III–V semiconductors more easily than between two group IV semiconductors(Si and Ge) [16].



ELLENBOGEN J. C., "A Brief Overview of Nanoelectronic Devices", 1998 Government Microelectronics Applications Conference (GOMAC98,) Arlington, VA, 13-16 March 1998.

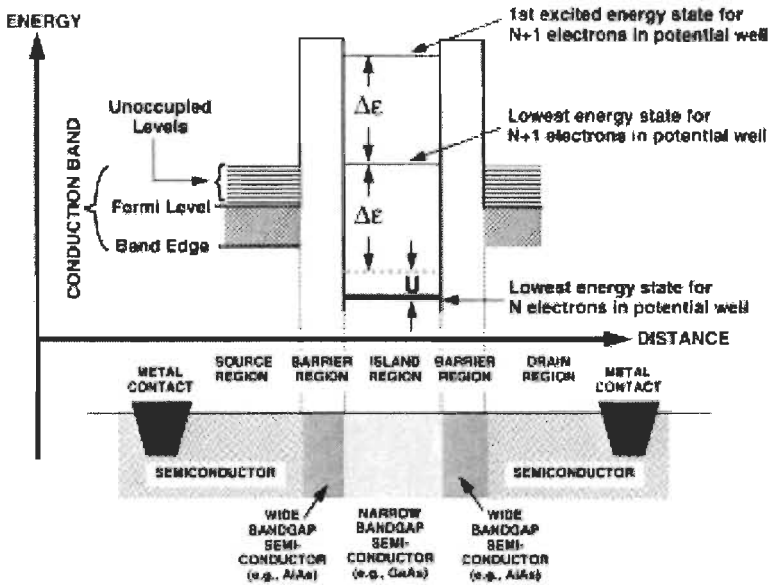
Figure 2-2 Nanoelectronic devices

Quantum effects in islands and potential wells: Islands being the switching area of a nanodevice have at least one dimension in the range of 5 nm to100nm. Island must be defined from the other regions in some ways. Many times the material that it is composed of is different from the surroundings in these cases the island is usually surrounded by the oxide of island's material or some other insulator. In some cases the material is not important and the boundaries of the island are shaped by electrodes put in a certain pattern. Anyway, what matters from an electrical point of view is that island is always separated from its surroundings by some potential energy barriers. In this way it can confine electrons within these potential walls [16].

One negative point for quantum effect and single electron solid state devices is that usually the region that mobile electrons can occupy inside the island is just a small puddle in the middle of the island. The reason is that these electrons are repelled by surface charges on the boundaries of island. So the island has to be physically much bigger than the confinement region. Around the puddle there is a large portion of island's space called depletion region which is not used efficiently. This limits the miniaturization potential of these devices [16].

There are two quantum mechanical effects that act on confined electrons and on nanoscale islands that need to be discussed. First one is quantum states. Each particle in general can only occupy one energy level among a finite (in the practical energy limits) number of discrete energy levels called quantum states. The amount of energy in each level and the distribution of these discrete energy states depend on the physical and electrical properties of the environment. In an island electrical potential inside the walls effects the amount of energy an electron has to have to occupy one of the states, available inside the island. The distance between the walls also has an inverse relation with the difference of energy between the states. So a smaller island has quantum states more separated in energy. The other important quantum effect is tunneling: Depending on the magnitude of the potential barriers, if the distance between them is small enough (5 nm to 10 nm), electrons with energy levels less than the height of the barrier, have a chance to pass through the barrier on enter or exit the island. Of course these two effects do not cancel each other. So for an electron to tunnel through a barrier, there must be an available quantum state on the other side with an energy level low enough for the electron to occupy it. Also not every electron that has a chance to occupy a state on the other side of a barrier will tunnel. Let's

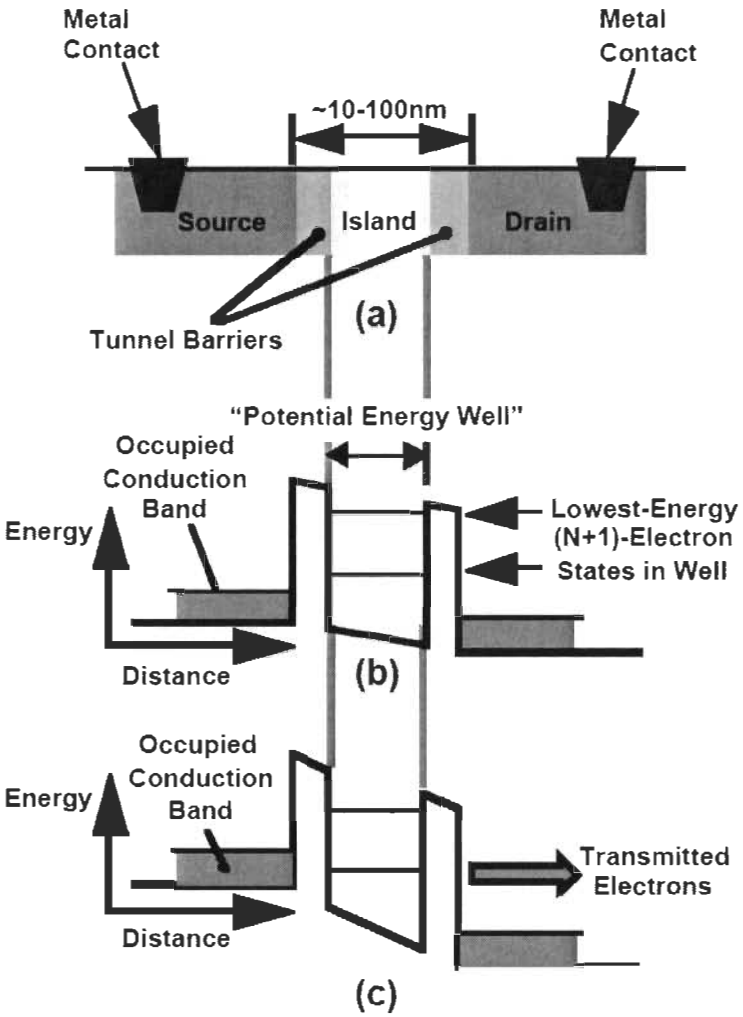
see how these effects work in the context of a solid state quantum effect device: Bias voltage, applied to source and drain, pushes the mobile electrons in the source conduction band to pass through the island and reach the drain with lower potential. The island is separated from source and drain by high potential barriers, so electrons need to tunnel into the island and then tunnel out of it and into the drain region. For that to happen, for an electron that occupies a state in source with a certain energy level there must be an available state inside the island with the same energy level so that the electron can occupy it after tunneling. The same goes for the tunneling from island to the drain. The difference is that because drain has a lower potential level, there is always a free state with energy lower than the energy of the electron inside the island. So once an electron get inside the island, the bias voltage can easily push it into the drain region. In bulk semiconductor because of the wide physical limits, there are much more energy levels placed close to each other, both in the source and in the drain. These energy levels form almost a continuous band. But in a nanoscale potential well the states are so discrete. (Figure 2-3) [16].



GOLDHABER-GORDON D., MONTEMERLO M. S., LOVE J. C., OPITECK G. J., ELLENBOGEN J. C., "Overview of Nanoelectronic Devices", Proceedings of the IEEE, vol. 85 , Issue: 4, pp. 521-540, 1997.

Figure 2-3 Quantum well of a RTD

Resonant Tunneling Devices: To turn a device on, the main idea is to lower the energy of the quantum states inside the island relative to the energy bands of source and drain so that electrons in the source band can enter the island. In the case of an RTD (Figure 2-4), this is done by increasing the bias voltage which puts some parts of source energy band equal to a free state in the potential well and let an electron from source occupy it [16].

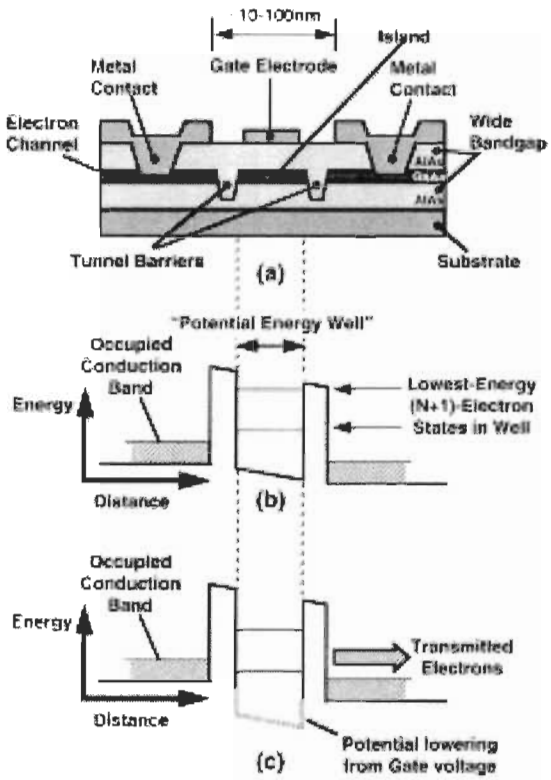


GOLDHABER-GORDON D., MONTEMERLO M. S., LOVE J. C., OPITECK G. J., ELLENBOGEN J. C., "Overview of Nanoelectronic Devices", Proceedings of the IEEE, vol. 85 , Issue: 4, pp. 521-540, 1997.

Figure 2-4 Structure of a RTT

If bias voltage is not high enough to create an overlap between source conduction band and lowest free energy band in island, no electron can pass into potential well and so there is no current passing. In this state the device is off or "out of resonance". [Figure2-4 (b)] If bias voltage is high enough to put an unoccupied state in island within the reach of an electron in source band, current passes and device is on or "in resonance" (Which indicates

the frequency associated to the energy level of potential well is in resonance to the frequency associated to an electron's energy in source).[Figure2-4(c)]So we have a device whose current is turned on and off according to a bias voltage. This is a two terminal resonant tunneling device or a resonant tunneling diode (RTD).This concept, further developed, leads to a three terminal device in which the relative state energies of source and island are adjusted by a third gate's voltage directly connected to the potential well. By controlling this voltage potential of states in the potential well can be controlled. So this is a device whose current is controlled by a small voltage. It is called a resonant tunneling transistor (RTT). [Figure 2-5] RTT is capable of replacing MOSFET, working as switch or amplifier [16].

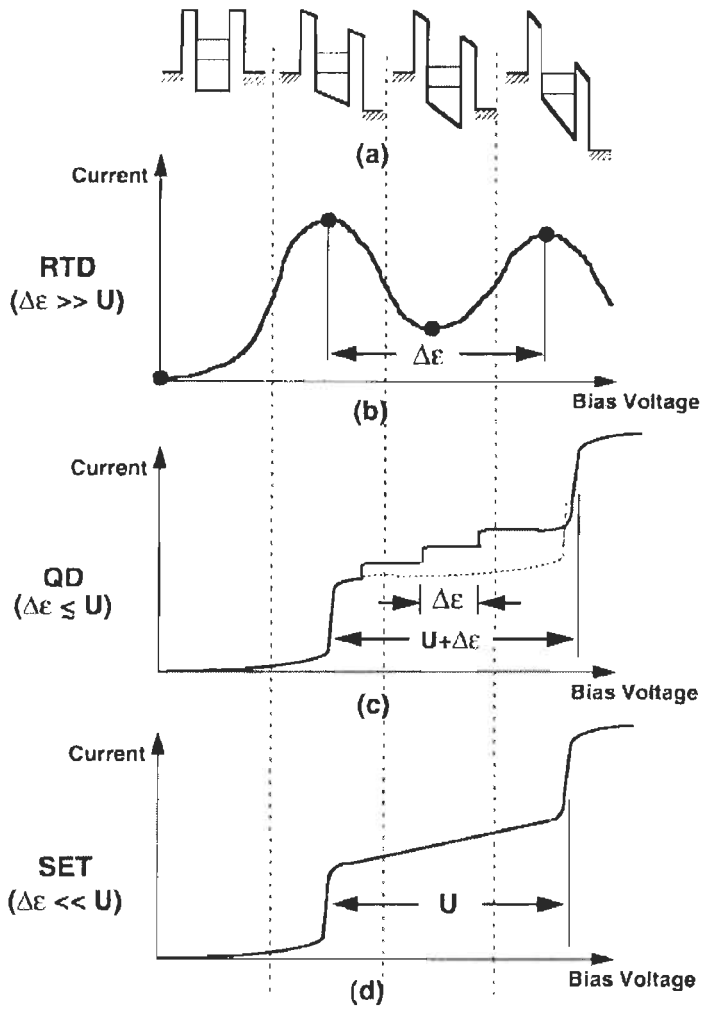


GOLDHABER-GORDON D., MONTEMERLO M. S., LOVE J. C., OPITECK G. J., ELLENBOGEN J. C., "Overview of Nanoelectronic Devices", *Proceedings of the IEEE,* vol. 85 , Issue: 4, pp. 521-540, 1997.

Figure 2-5 Structure of an RTT

We know that there is more than one free quantum state inside the island. Now if the energy difference between these states is bigger than the difference energy between source band edge and Fermi level, in other words if source conductance band is thinner than distance of island states from each other, for each time that source conductance band reaches the level of a free state in well, a flow of current will happen. So RTDs and RTTs are capable of multiple on and off states. This valuable feature means that a circuit can be implemented with fewer devices using RTDs and RTTs rather than MOSFETs. Because with multi state switches a function is realized with less switches. This leads to fewer

devices thus less heat generation. In Figure 2-6 (b) for a case where conductance band is much thinner than potential differences in well, the current peaks represent the moments when bias voltage (or in the case of RTT, the gate voltage)aligns a free energy state with the conduction band. As shown in Figure 2-6 (c), even when conduction band is wider than state energy differences, a step pattern is formed that might work as a multi-state logic [16].

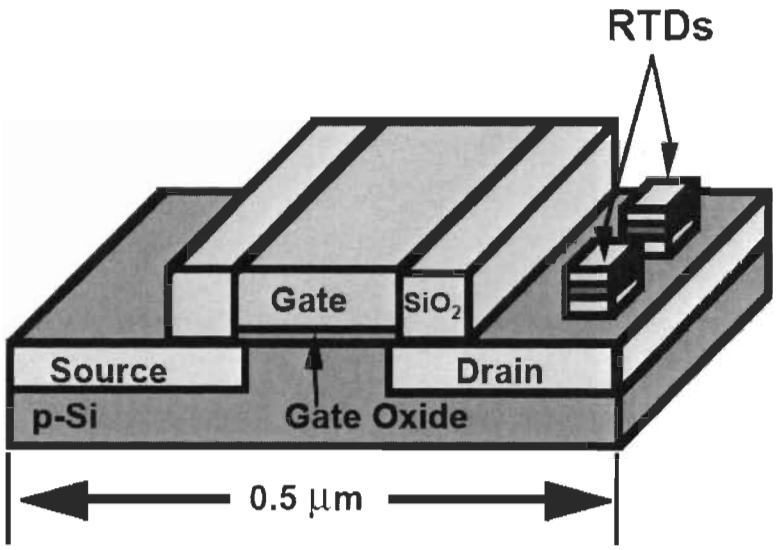


GOLDHABER-GORDON D., MONTEMERLO M. S., LOVE J. C., OPITECK G. J., ELLENBOGEN J. C., "Overview of Nanoelectronic Devices", *Proceedings of the IEEE,* vol. 85 , Issue: 4, pp. 521-540, 1997.

Figure 2-6 Switching of solid-state nanoelectronic devices

To fabricate these devices layers of different III/V semiconductors (for example GaAs and AlAs) are put together to form the source and island and drain. To make a resonant tunneling diode [Figure 2-4(a)], two insulating barriers are used to envelope the island semiconductor and create the potential well between them. Islands in RTDs are about 10 nm wide [17].

Hybrid Microelectronic-nanoelectronic Resonant Tunneling Transistors: Fabrication of RTTs is still a technological challenge. They are too small, too complex and too sensitive to be produced uniformly and efficiently with current technology. There are devices designed for the period of transition from microelectronics to nanoelectronics that employ the advantage of multi state logic of quantum effect devices in the current MOSFETs. If RTDs are put into drain or source of a micro scale MOSFET an RTD-FET or a hybrid RTT is built whose source-drain current has multi on/off states controlled by RTD's bias voltage.(Figure 2-7) In this way the logic density of circuit increases without scaling transistors or one can say that the current high density of chips that is producing problems can do more in this way [16], [17].



GOLDHABER-GORDON D., MONTEMERLO M. S., LOVE J. C., OPITECK G. J., ELLENBOGEN J. C., "Overview of Nanoelectronic Devices", *Proceedings of the IEEE,* vol. 85 , Issue: 4, pp. 521-540, 1997.

Figure 2-7 A hybrid RTD-FET

Anyway, on a larger time scale hybrid logic is mostly a practical step on along path that ends in production of purely nanoelectronic devices [16].

Single electron transistors: Single electron transistors are very similar to RTTs. They also have a potential well, isolated from source and drain. The difference is that here the island is so small that it can contain only a few electrons at the same time. In fact electrons in a process called Coulomb blocking repel the other electrons and stop them from entering the limited space of the island. So there is a balance between the force of bias voltage and the force of Coulomb blocking and no current passes. Using a voltage induced into an electrode that acts as the gate in RTTs, potential of the island can be increase so that one electron from source can enter island. Then one electron will leave the island and go to the drain; hence electrical current flows. Single electron transistors can only operate at low

temperatures because high thermal energy will destroy the order of this very sensitive device and will make electrons jump over the well in their high energy resonances, regardless of Coulomb blocking [15].

Chapitre 3 - High Performance Computing Systems

Conventional chip–package thermal analysis techniques have been so slow that evaluating numerous design alternatives was prohibitively expensive during IC design. As a result, most thermal optimization was done during packaging and cooling solution design. Unfortunately, by that time the design is already tightly constrained. Recently, a number of researchers have developed fast thermal analysis techniques for use during the IC design process. Using these methods, heat transfer through chip and cooling package is modeled using the classical Fourier transport model [14].

Although some of these techniques are fast enough for use during IC design and within run-time thermal management techniques, they are all based on the Fourier heat flow model. This model cannot capture phonon quantum thermal effects and yields inaccurate results when used at length scales on the order of phonon mean free path. These observations are supported by the data presented in Section IV-A. Techniques with different fidelities and efficiencies have been developed to model nanoscale device-level phonon heat transport, including molecular dynamics methods, Boltzmann transport equation (BTE), and ballistic-diffusion model. Computational complexity has been the primary challenge of adopting nanoscale heat transfer methods for large-scale IC chip–package thermal analysis. Molecular dynamics methods model heat transfer by directly simulating inter atomic interactions. Approaches implemented using these methods are highly accurate. However, they are extremely computationally expensive. The BTE method

and its variants model heat transfer by simulating the transport of phonons. It can accurately approximate ballistic phonon transport. BTE methods are much more efficient than molecular dynamics methods. However, their computational complexity remains prohibitive, and their use has been restricted to device-level analysis. The ballistic-diffusion based thermal analysis method is an approximation of the Boltzmann transport method. Although the ballistic-diffusion model is the most efficient of these, it is still much more computationally-demanding than the Fourier model. In addition, results from the ballistic-diffusion model tend to have low fidelity. In summary, there is a gap between the efficiency and accuracy of nanoscale and chip-package thermal analysis techniques that must be closed if high-quality temperature-aware design techniques and run-time thermal management algorithms are to be developed for ICs composed of nanoscale devices [14].

The study of nanosystems has been the subject of several searches in the last decade. But the works related to the heat transfer of these new components is still at the stage of development [15].

So, heat analysis, an increasingly important field in order to improve design, demands a lot of computation. A solution to this problem can be employing parallel algorithms using MPI standard and implementing them on high performance computing systems [3], [15], [18].

3.1 Parallelization

High performance computers (HPC) distribute the work between many processors which are put in a certain structure which is unique to each HPC facility (CLUMEQ, West

Grid...). By dividing the job between many processors and making them work in parallel, the computation time needed decreases dramatically. In fact if n processors work in parallel, the processing time is ideally divided by n. Also, the memory available for the whole process is n times the memory of each processor. What needs attention in parallelization is how to distribute a job between nodes and how to transfer the data between them during the processing. Message passing Interface (MPI) is employed to create efficient connections between these processors [15].

According to Flynn's taxonomy [19], there are four processing unit architectures imaginable to do a task. In Single Instruction Single Data stream (SISD) one processor performs one series of instructions on one set of data. This architecture is used in the traditional uniprocessor PC (not PCs with multi-core processors). A single Instruction Multiple Data stream (SIMD) is when one instruction is given to more than one processor to be performed on separate parts of data simultaneously. So parallelism (on processed data) is possible in this architecture. In Multiple Instruction Single Data stream (MISD) machines different instructions are given to different processors to be performed on a single piece of data. Here also a parallelization (this time on instructions) is formed. This is an uncommon architecture which is ideal for applications where fault tolerance is low. Because different processing units work with the same data and they have to be in good agreement on its correctness. So each error can be discovered by other units. Multiple Instruction Multiple Data streams (MIMD) gives different instructions to different processors and they perform them in parallel on different data. MIMD can be implemented in two forms: Single Program Multiple Data (SPMD) in which one program is executed independently on different processors and Multiple Program Multiple Data (MPMD) in

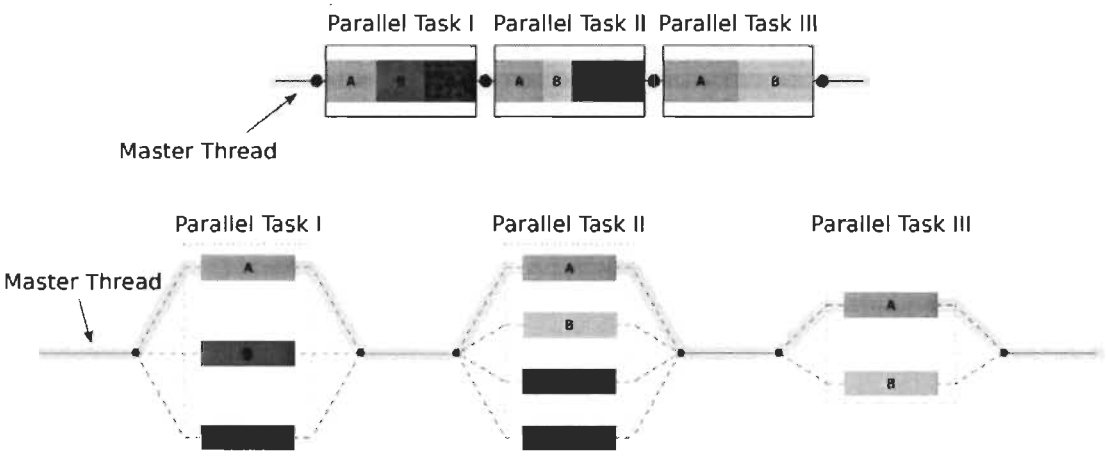
which different programs are executed on different processors at the same time. Usually in these applications one processor executes a "manager" program which communicates with other processors executing another program. Each processor sends its results to the manager independently and there the results are fed into the manager program [20].

SPMD architectures are very common and are realized in two forms: shared memory and distributed memory. In shared memory architecture, processing units have simultaneous access to one shared memory. So they can pass information by writing them on memory and no separate communication between them is needed. So they are fast, but instead they are more expensive and in programming level attention must be paid to avoid any conflict of tasks done by processors. In distributed memory architecture each processor has its own memory. Processors communicate with each other directly in order to pass information [15].

There are two SPDM algorithms, OpenMP and MPI Standard, developed respectively for shared memory architectures and shared and distributed memory architectures that are introduced here.

OpenMP: Open multi-processing is a programming interface including some directives, libraries and variables that helps create a parallel code and execute it in C, C++ and Fortran languages. With OpenMP, a task, that has some elements that can be executed in parallel, is coded as a parallel algorithm. To do this it is desired to follow the master thread of the code and fork it to a number of threads whenever a parallelizable element is reached. These threads can be executed separately. OpenMP gives each of these threads an ID number and sends them to different processors. When the parallel task is finished, the master thread continues to run until it reaches another parallel task (figure 3-1) [20] .

MPI Standard: Message Passing Interface (MPI) is a programming interface to create clusters of processors, placed in a HPC facility or in distant PCs, for parallel computing. MPI does that by managing the communication between the processors. MPI works with both shared memory machines and distributed memory machines [20].



wikipedia.org

Figure 3-1 OpenMP schematic in comparison to single processing

3.2 Infrastructure

CLUMEQ cluster is used to implement the parallel code created using the NanoMOS 3.5 MATLAB code. CLUMEQ is a part of Canadian HPC infrastructures gathered under the name Compute Canada.

Compute Canada is national collective of consortia of HPC infrastructures existing all over Canada. These infrastructures are geographically distributed aver the country and cover the high performance computation needs of researchers, not in a regional sense, but

in a country-wide way. Although these infrastructures are located in seven different regions, they are joined together in Compute Canada as a net of distributed infrastructure accessible through one gateway. So having an account in Compute Canada, offered to researchers, professors, students..., gives you the right to execute your code on any of the consortia and also benefit from technical support, educational programs and other services offered by them [21].

These consortia are: ACEnet (Atlantic Computational Excellence Network,) CLUMEQ (Consortium Laval, Université du Québec, McGill and Eastern Québec), RQCHP (Réseau québécois de calcul de haute performance), HPCVL (High Performance Computing Virtual Laboratory, SciNet (based at University of Toronto), SHARCNET (Shared Hierarchical Academic Research Computing NETwork) and WestGrid (Western Canada Research Grid). Each of them includes some HPC infrastructures or supercomputers based in universities placed in that region. Compute Canada cooperates with these HPC consortia to plan future infrastructures and fund these projects, to provide the software and hardware and to provide technical support for the researchers in each regional facility [21].

CLUMEQ includes two supercomputers situated in Mcgill University, Montreal and Laval University, Quebec city. These two are:

Colosse: A cluster of 960 nodes (8 processor cores and 24 GB of RAM each), with totally 7680 cores and 23 TB memory, with a full bisection Infiniband QDR network and a Lustre parallel file system of capacity up to 1 PB (currently 500 TB) [22].

Krylov: A cluster of 48 nodes (4 or 8 processor cores and (300 cores total) and 8 or 16 GB of RAM each), with totally 300 cores, with Infiniband SDR network [22].

Supercomputers are a great and central point to this study. One very important aspect of a supercomputer is its structure, both electronically and mechanically. Computation power of a supercomputer is a direct result of its number of nodes and the power of each node. So, in respect to electronic hardware, the model, number and other properties (for example speed and memory) of processors used are the first features of a supercomputer to mention. Of course the data transfer cables and other hardware used in the system must have a capacity that covers the needs of the system and lets it work to its full computation power. Choosing and employing software is another concern in managing all these processors work in parallel. Bellow a structural discussion of Colosse is presented to cover these hardware and software aspects. On the other hand, when the number of processors in a cluster is huge, the problem of heating becomes huge too, because processors are the device in a computer that produces the most heating. So when there are a lot of them in a computer, heating goes to a whole new level. Surprisingly extensive and expensive measures have to be taken to exit the heat in a supercomputer. To do that, a special cooling system has been used for colosse that forms the mechanical part of the structure. The size and power of these cooling devices is even greater than that of the electronic parts. This fact reinforces the necessity of studies of this kind and should be discussed here. So a description of the cooling system of Colosse will also be a part of this chapter [22].

As mentioned above, Colosse has 960 computational nodes. Each of these nodes includes 2 Intel Nehalem-EP processors each of which has 4 cores and 24 GB RAM. Colosse also has 40 nodes for infrastructural responsibilities. So in total there is 8000 cores

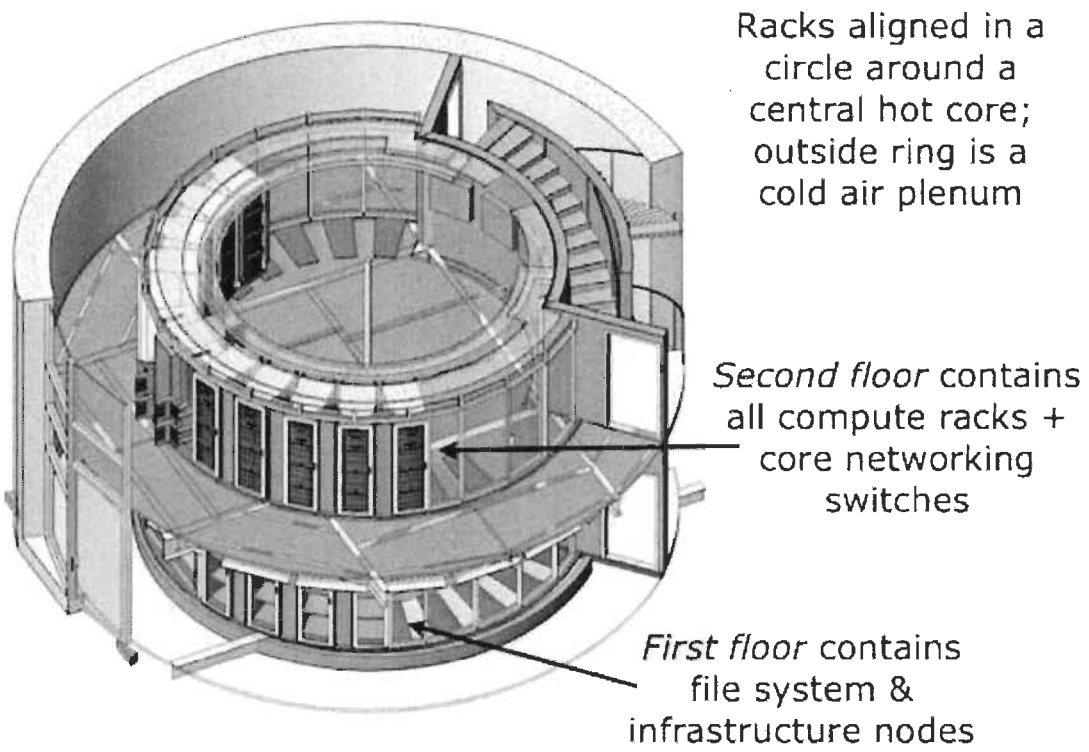
and 24 TB RAM. The nodes are connected with infiniband QDR cables with a speed equal to 40 Gb/s. The infrastructure nodes are connected with an ethernet connection to each other and to the outside world through web. Half of them make a parallel file system of 1 PB capacity [22].

Infiniband is a certain kind of communication system that connects nodes mostly in HPC applications. It provides failover capability, which means in the case one device stop working it can switch to a stand by one. Infiniband link is also scalable. It is usually used to connect the processors to I/O and which in this case is mostly the RAM. Infiniband uses a switched fabric model in which the nodes are connected to each other through one or more routes. This is contrary to the broadcast networks which have a hierarchical model. SDR infiniband sends data one every clock cycle. QDR infiniband, on the other hand, is an infiniband that sends data four times in a clock period (on the rising and falling edges and between them.) So it is 4 times faster than an SDR infiniband. To find the moments where the clock is in 90 and 270 degrees it uses another clock with 90 degrees delay in relation to the main clock [20].

Lustre is a special file system developed by sun microsystems. File system is a method of storing and managing files in a computer. Lustre, made from mixing words linux and cluster, is meant to service clusters with as much as tens of thousands of nodes and some PB of memory and some hundreds of GB per second data transfer rate and at the same time have a high availability. This file system is also scalable. It is now widely employed in supercomputers [20].

Colosse is assembled inside the remains of a particle accelerator in University of Laval campus. The particle accelerator was first built in the 60s. Lately after the colocsse project was started, in search for a place to set up the supercomputer, it was decided to use the former accelerator for this purpose. The final design was a very exiting plan: The old particle accelerator had a big silo that was abandoned for years. The plan was to turn this structure into the home of new supercomputer [22].

A three floor metal structure has been built inside the silo. Cabinets containing processors are put on these floors in a circular form so that they make a smaller cylinder inside the big cylinder which is the inside space of the silo. The first floor is where the infrastructure devices and servers connecting to the outside world are placed. Calculating processor cabinets are placed in the second and third floors. By this three floor design; accessible area to place the processors has almost tripled. So now there are a lot of extra space in the second and third floor to place more processors in future upgrades of the system (figure 3-2) [22].

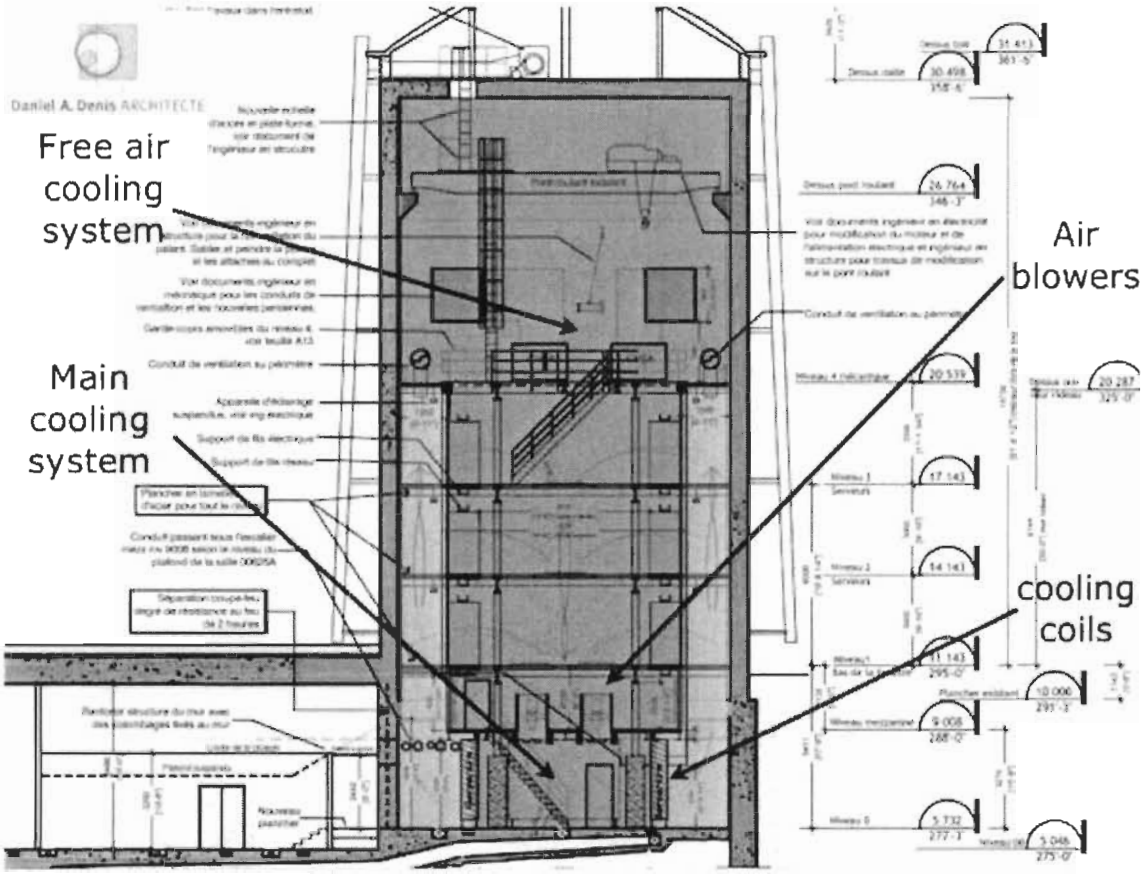


CLUMEQ (www.clumeq.org)

Figure 3-2 Placement of the electronic devices

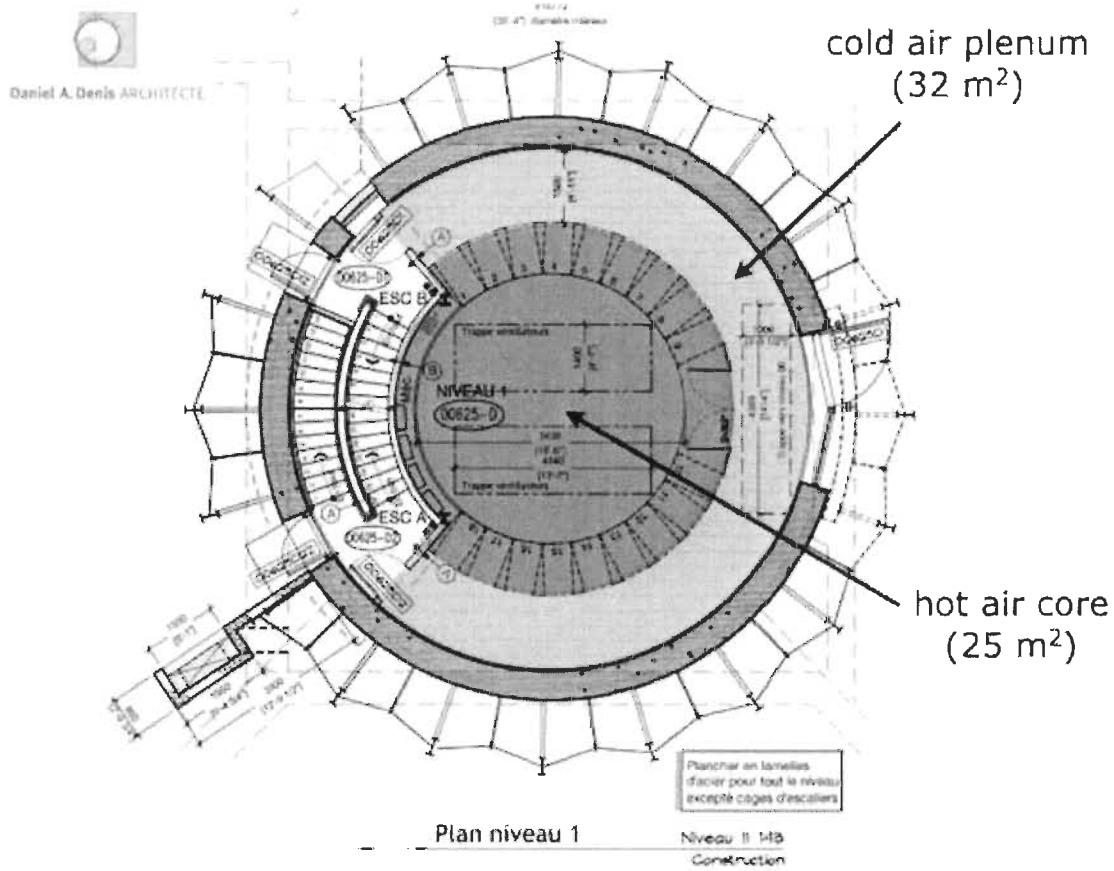
The circumference of this smaller cylinder is isolated from the space between two cylinders so that a warm tunnel of air can be formed inside it to exit the heat. This tunnel is fed by big fans placed bellow the small cylinder that blow the air from hot air tunnel into the cooling radiator in the ground floor and then to the cold air area between two cylinders. In other words these fans produce a positive pressure between the cold air region and hot air region so that the cold air passes through processor cabinets into the hot air tunnel. By dividing the inside of the silo into two hot and cold areas, the efficiency of use of fans increases, because power of fans is not used to blow hot air to the processors. So it is possible to use fans with less power or less speed of cooling air in the design. Also

distributing the cabinets in three floors (in comparison to putting all of them in one floor) increases the surface of cooling and makes it more efficient. The circular placing of the cabinets decreases the length of the cables too. The circular structure of the silo itself and circular placement of processor cabinets are ideal for the flow of the cooling air, because they do not have any corners, as a square structure would have had (Figures 3-3 and Figure 3-4) [22].



CLUMEQ (www.clumeq.org)

Figure 3-3 Side-view of the structure



CLUMEQ (www.clumeq.org)

Figure 3-4 Structure viewed from above

In addition to the main cooling system, which is in ground floor (beneath fans) and cools the hot air before it goes to cool air region, there is a system to use the cold air from outside to do part of the cooling process. Also in the cold season the heat produced by processors is employed in heating system of the campus. All this shows the great amount of heat produced by what we would have carelessly called "just some electronic devices" [22].

In total, there is 56 cabinets on the metal structure in three floors which are cooled by a system with a ventilation capacity of 132,500 CFM (62,500 liters per second) and a cooling capacity of 1.2 MW [22].

In this section we will review the steps that need to be taken to connect to the colosse supercomputer and login to the system and then we will see how to start a new project and how to send and have your job executed on the system. First of all a researcher needs to make an account on Compute Canada website. Students can do that through the account of their supervisor and using the information given to them by their supervisor. Next, an account is created on the certain consortia chosen by the researchers and that lets them apply their job on the supercomputers within that consortia. Here we suppose that one has created the necessary accounts to access Colosse supercomputer and now wants to login to the system and use it to execute a parallel code [22].

3.2.1 Accessing Colosse:

Login: Users do not have direct access to the computing nodes. They have to connect to the head node through a secure shell client. A shell client is a program working as an interface between a user and an operating system. Referring to the outer space of something, the word "shell" is also generally used for cases in which a program plays the role of interface to a system or program. For example a shell account is an account that a user has on a remote server and can access some data, memory or applications through that. A secure shell is a protocol used to make a secured and encrypted connection between two points. It can be used for example to connect to a shell account. Secure shell (SSH) was created to replace connection protocols like telnet that do not encrypt the data [20].

So as was mentioned above users connect to the head node through an SSH client. Here they need to enter their user name and password, defined when they created their accounts. In Linux and MacOS systems this is done easily by typing from a terminal [22]:

$ ssh

user@colosse.clumeq.ca

A Windows user has to install some software to makes this SSH connection possible. These programs are accessible for free on the web. Some of them can be found in the following address [22]:

http://www.openssh.com/windows.html

Then it might be necessary to create a profile using the interfacing program installed. To do that these information are needed [22]:

Host name: colosse.clumeq.ca

User name: your user name

Port number: 22

File transfer: Now that the connection to colosse has been realized, user needs to use a secure protocol to send and receive data and files. Secure Copy (SCP) and SSH File Transfer Protocol (SFTP) are two protocols that could be used. These two are designed based on SSH, but they are compatible with other protocols too. Their job is to guarantee a safe transfer of files. SCP is just a means of transferring files, but the newer SFTP is more advanced and is also capable of doing some operations on the distant files too. Some of its advantages over SCP are the possibility of resuming a transfer after it has been interrupted,

and removing files remotely. Windows users need to install a program (examples: FileZilla and CoreFTP) to add these protocols [22].

This is the File system structure accessible to users when they connect to colosse:

Table 3-1 Data storage in colosse

www.clumeq.org

Path	Description	Quota
/home	Your account path	5GB
/rap	Group shared space. Holds data or common tools for the group	100GB
/scratch	Temporary scratch space. Tuned for fast I/O.	

Compiling: There are different softwares accessible on colosse that users can call them and use them to compile their codes. This is done using the module command. Module is a command that make the shell ready for a curtain application. For example it creates the necessary variables and so on. In brief users call their desired application using this command [22].

Here some useful module commands are mentioned that are taken from clumeq.ca:

"Show all modules available on the system:

$ module avail

Show currently loaded modules:

$ module list

Load a module (This command is also used inside your Grid Engine submit files):

$ module load <*modulename*>

Unload a module:

$ module unload <*modulename*> " [22]

In the following example, again taken from clumeq.ca, the first line calls a compiler already existing on Colosse and the second line calls a simple C program:

"[USER@cyclops ~]$ module load compilers/intel/11.1.059

[USER@cyclops ~]$ $CC -o hworld /clumeq/example/hworld.c" [22]

Running the code: To run a code on Colosse one should send it to Grid Engine (GE). There it will enter a queue and later will be fed to the cluster by GE [22].

These are some commands, taken from clumeq.ca, to work with GE:

"**qsub***script_file* - submits a new job, where *script_file* is a text file, containing the name of your executable program and execution options. Examples of *script_file* are explained below.

qstat - shows a current list of submitted jobs. Can be used with the following arguments:

qstat -ext - shows a listing of all your queued and running jobs

qstat -g c - shows used and available resources for each queues

qstat -u "*" - shows jobs information for all users

qstat -j*JOB_ID*- shows details for a specific queued or running job

qstat -t - shows the resources used by your running jobs

qdel*job_ID* - kills the job, or removes it from the queue. The *job_ID* can be obtained by qstat command.

qconf -sql - shows existing queues in the system" [22]

Here the structure of a script_file is explained and then the actual command in command line is mentioned. (From clumeq.ca) [22]:

"#!/bin/bash
#$ -N JOB_NAME
#$ -P PROJECT
#$ -l h_rt=00:05:00
#$ -pe mpi NB_CORE
module load SOME_MODULE
/your/program/path/here

The statements above have the following meaning:

- The name of your job.

- Your project name which is the CCDB RAP id. The format is abc-000-00.

- Estimated run time of this job. For example, 5 minutes is 00:05:00

- Parallel environment (PE). See the description of the various PEs below.

- Commands to be executed on the compute node :

$ qsub sub.sh
Your job 212 ("MyJob") has been submitted" [22]

There are three parallel environments (PE) accessible on Colosse. As said above, the type of parallel environment chosen for the code to be executed in must be written as part of the submit command. These environments and their explanation and then the way they are addressed in a submit command and also the scipt_file referring to them is taken from clumeq.ca [22]:

"mpi
"-pe mpi" is used for mpi job. Those processes can be run anywhere on the machine using a "fill up" allocation rule (all slots allocated in a node before moving to the next one). "nb_slots" represents the number of cores requested.

node
"-pe node" The processes will run on compute nodes dedicated to this job. Your "nb_slots" (number of core) should be divisible by 8.

smp
"-pe smp" is for non mpi jobs and mulithread jobs. This parallel environment allocates cores (slots) on a single node. The maximum number of slots (nb_slots) is 8" [22]

"#$ -pe mpi NB_CORES
#$ -pe node NB_CORES
#$ -pe smp [NB_CORES]" [22]

"#!/bin/bash
#$ -N MyJob
#$ -P abc-000-00
#$ -l h_rt=00:05:00
#$ -pe mpi 32
module load mpi/openmpi/1.3.4_intel
mpirun /your/program/path/here" [22]

Chapitre 4 - Modeling

4.1 Physical and mathematical modeling

Semiconductor industry has been constantly developing in many different directions for many years. These improvements, according to their area of effectiveness, can be categorized into some trends of development called scaling trends: integration level, cost, speed, power, compactness and functionality. Most of these trends are the result of constant reduction of minimum feature size in semiconductor processes. Moore's law is a well-known example of scaling trends: number of components on a chip is doubled every 24 months. But still, scaling is not necessarily identical to feature size reduction in fabrication processes. For example now there is a tendency towards equivalent scaling, as opposed to geometrical scaling (which is basically the feature size reduction, resulting in Moore's law). Equivalent scaling is the improving effect of new materials, better designs, 3D integration, more efficient processes and even better software, benefitting the industry just as geometrical scaling has been doing [23].

Having made this distinction, we can return to the subject of this study. Transistor scaling is mostly, but not only, a result of feature size scaling. Development of materials (like high k insulators), designs and architectures (SOI, thin-body and multi-gate MOSFETS) and new fabrication processes have great effect on transistor scaling too. Here, the importance and possible limits of the scaling will be discussed, in regard to performance, geometry, material and process issues.

ITRS predicts that physical gate length in MOSFETs will reach 10 nm in 2020, but at that scales difficulties are expected to appear that will challenge the whole CMOS scaling future. The effort needed to stretch current CMOS technology through deep scales is becoming more and more costly. The costs of production has been increasing in a rate that there are questions whether it can continue in the same way for more than 15 years. It seems necessary to look for post-CMOS devices that improve device performance and cost per function. It is worth mentioning that part of this development load will be carried by new design and manufacturing paradigms (currently, performance of processors does not improve equal to the increase in the number of their transistors) and also by integrating technologies like Sip and SOC. Without doubt these new axis of development will change concept of computation systems, but in the long run, transistor technology has no way but to improve. So the move toward enhanced CMOS architectures and ultimately post-CMOS devices must be taken seriously. These devices can be quite similar to today's CMOS, only with new materials and designs, or they can be radically different, like quantum and molecular transistors. In any case, studies that target the edges of CMOS technology and apply new models to simulate them will help, either by helping scaling CMOS to its maximum potential or by creating an understanding of models that might describe future devices. Noting that in gate lengths of less than 10 nm non-classical effects will not be negligible; it seems a necessary choice to try to employ ballistic and semi-ballistic models in those lengths, as it is the goal of this project [23].

Conventional scaling of CMOS consists of reducing gate length, gate dielectric thickness and increasing channel doping. In planar MOSFETs high channel doping helps reduce short channel effects, but it also reduces mobility and increases leakage power

consumption. The main function of multi-gate or ultra-thin body fully depleted SOI solutions is to fight these effects and improve performance.

Off current, consisting of gate and drain leakage currents, grows 10 times in each generation of CMOS. Leakage power has exponential relation with gate length, oxide thickness and threshold voltage. So as generation appear, a greater portion of total power leaks. This, supposing a needed computation power level, increases total power. Today almost 50% of total power is leaked [23].

Some challenges ahead of semiconductor industry and in some cases their respectable solutions were reviewed. In the following sub-chapters, this discussion is extended to a materials point of view and today's difficulties and future solutions are discussed. Qualitative and mathematical description of MOSFET is employed to find and discuss the main parameters in structure of a MOSFET that have impact on its performance. Improvement of these parameters is the focal point of industry in transistor scaling process. Realization of an actual MOSFET is studied. First, a fabrication process, designed for the specific choice of device in this study, is presented in details. Then packaging possibilities are discussed and some suggestions are made for an actual packaging process. At the end to make an actual design feasible, a series of simulations is performed to study the parameters that play a role in design of highly scaled MOSFETs.

4.1.1 Material considerations

Metal gate transistors beyond 16 nm technology, will need an Equivalent gate Oxide Thickness (EOT) of 0.6 nm. EOT is the thickness of a silicon dioxide layer having the same effect as a layer made from a high k material. Providing these EOTs will be difficult, even

with very high dielectric constant insulators. At the same time, channel mobility is decreased because of interface effects between silicon and high k material. Interface between high mobility channel materials (Ge, SiGe and III-V materials) and high K dielectrics is more problematic than a Si-high k interface, because it these materials have more complex surfaces and that they do not have oxides that can function as insulators [23].

Interconnect scaling has its own set of problems and solutions. High conductivity metals and low permittivity dielectrics and low k dielectrics (to reduce parasitic capacitance and heat flow and increase speed) are needed to make the connectors smaller and the isolation thinner [23].

Of course fabrication processes play a crucial role in transistor scaling. Apart from the obvious fact that new processes make minimum feature size reduction possible, use of more efficient materials or designs might impose innovation in process. For example, to fight short channel effects it will be helpful to create a channel with a carefully chosen doping profile, instead of doping the channel evenly. Less doping near the substrate insulator and also near the source and drain interfaces and more doping in the middle of the channel is desirable. To be able to do this, a selective epitaxy of channel with Si, Ge and C is performed so that doping can be varied locally [23], [24].

To achieve higher drive current, high transport materials can be used in channel: Ge on Insulator, III-V, nanowires, carbon nanotube [23].

4.1.2 MOSFET theory and design

MOSFET is a voltage controlled current source. Gate voltage controls the current in a channel made of semiconductor placed between drain and source electrodes. What are the

parameters that affect its function? To give an answer, more discussion of its structure is needed.

In an n-MOSFET, channel has p-type dopant and drain and source have n-type dopants. A voltage difference is put on drain and source electrodes but channel, even though it is doped, is not a conductor and current does not pass. Gate electrode is place on top of the channel and is separated from it by a thin layer of insulator material, so that it has no electrical connection with channel. Now if a positive voltage is given to the gate, free electrons in channel are absorbed towards gate. So, a region is formed inside the channel, just under the insulator where electrons are gathering. If the gate voltage is high enough, there will be enough free electrons gathered in that region to be able to conduct current from source to drain [25].

The amount of electrons gathered in the channel determines how easily current flows. The insulator put between gate electrode and doped channel make a capacitor whose electric charge is calculated from equation 01. Concentration of electrons is represented by Q. We see that gate voltage determines Q. So gate is controlling channel conductance and thus the current. Another conclusion that can be made is that by decreasing capacitance, C, MOSFET can be turned on more easily and it can produce higher drain currents. This, of course, is very important in scaling of MOSFETs. We will discuss this effect in more details [25].

Let's look at channel and see it as a resistor (R_{on}) through which drain current passes. As gate voltage increases, more electrons join the channel and it conducts better, so resistivity of the channel decreases. In I_d-V_d graph, this means that for higher gate

voltages, slope increases. Other parameters that influence R_{on} are length of channel and its width. It is obvious from ohmic resistance equation that R_{on} (slope of I_d-V_d graph) increases by an increase in length or a decrease in width of channel. If we draw a graph showing Id for different Vg s, we see that as explained above, by increasing V_g, I_d (for a constant V_d) increases. This rise starts from the point where electron channel is formed and device is turned on and continues until saturation. If R_{on} is decreased (by a decrease in length or an increase in width), for a certain V_g, Id will be higher, so the slope of the graph between turn on point and saturation increases. Semiconductor industry has always tried to minimize channel length to reach higher drain currents. But the same does not go for width. More width can be reached easily. In the easiest solution two MOSFETs are connected in parallel, doubling the width. But this increases total gate capacitance (general capacitance equation) which might limit the speed. So a tradeoff is necessary here. Another parameter of equal importance as channel length is gate insulator thickness. A thinner insulator means that the capacitance between channel and gate is bigger, leading to more charges gathered in channel. We now that it means better channel conductance and less R_{on}. It is an established trend in semiconductor industry to decrease insulator thickness to reach higher drain current, just as it was for channel length [25].

Current flow in channel is a drift current. It means that carriers are accelerated by an electrical field, so there is a statistical velocity of carriers that means a displacement of charges and flow of current (the other form is diffusion in which carriers move from high concentration region to low concentration region by diffusion). This is shown in (4-1). This velocity, along with charge Q, in (4-2), yields to drain current. The charge is calculated

from (4-3) (pinch off considered). In this way Drain current is calculated for device's terminal voltages as (4). When (4-5), channel works in its linear region and (4-4) can be linearized to (4-6). It is in this region that R_{on} can be defined. This equation formulates all above discussion [25].

$$v = -\mu_n E = +\mu_n \frac{dV_{(x)}}{dt} \tag{4.1}$$

$$I = Q.v \tag{4.2}$$

$$Q_{(x)} = WC_{ox}[V_{GS} - V_{(x)} - V_{TH}] \tag{4.3}$$

$$I_D = \frac{1}{2}\mu_n C_{ox}\frac{W}{L}[2(V_{GS} - V_{TH})V_{DS} - V_{DS}^2] \tag{4.4}$$

$$V_{DS} << 2(V_{GS} - V_{TH}) \tag{4.5}$$

$$I_D \approx \mu_n C_{ox}\frac{W}{L}(V_{GS} - V_{TH})V_{DS} \tag{4.6}$$

By increasing V_d , Id increases. But MOSFETs reach saturation when V_d is higher than overdrive voltage (V_g - V_{th}). This happens because of pinch-off effect. Anyway, MOSFETs can be used in saturation region with a drain current that can be controlled by gate voltage. The saturation current is another important parameter in MOSFETs and can be calculated as in (4-7) [25].

$$I_D = \frac{1}{2}\mu_n C_{ox}\frac{W}{L_1}(V_{GS} - V_{TH})^2 \tag{4.7}$$

Table 4-1 MOSFET equation parameters

Symbol	Quantity
v	Speed of electron
μ_n	Mobility of electron
E	Electrical field
$V_{(x)}$	voltage in point x
$Q_{(x)}$	Charge in point x
W	Width
C_{ox}	Capacity of oxide
V_{GS}	Gate-source voltage
V_{TH}	Threshold voltage
I_D	Drain current
L	Channel length
V_{DS}	Drain-source voltage

4.1.3 Transistor structure

SOI transistors are the current answer to the scaling problems of MOSFETs. This architecture can be made using two different operation modes: partial depletion and fully depletion.

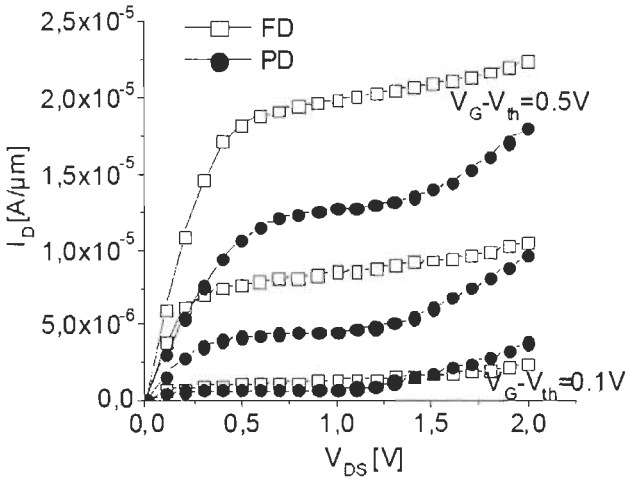
In a brief article [4], which is a primary presentation of an extensive study, these two modes are compared using different characteristics of transistors. The following plots, taken from reference [3], are based on fabricated models with gate lengths down to 25 nm [26].

The goal is to compare the benefits of these two types of transistors and in this way introduce the concept of dully depletion in transistors, which is employed in the modeling and simulation tool (NanoMOS 3.5) used in this study. In this article a lot of fabrication issues are explained which might be useful in the following weeks of our course.

First let's review the depletion concept. In a MOSFET gate electrode is isolated from the channel by an insulator. If for example the channel is negatively doped, then inducing a negative voltage in the gate will repel the electrons from the region near to the gate. This means that carriers cannot use a portion of the area of the channel. In this way channel current is controlled by gate voltage. These transistors are also called depletion mode MOSFETs [27].

Transistors with buried oxide of 100 nm and substrate thickness of 45 nm and 3nm oxide layer under the gate have been fabricated. Devices have n-doped channels and undoped channels for PD and FD modes respectively.

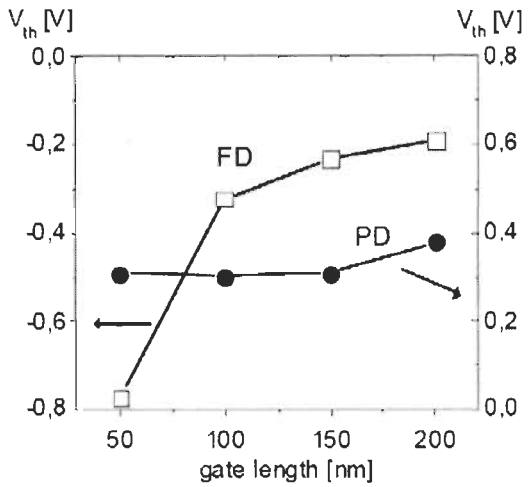
In Figure 4-1 output characteristics of a 200nm gate length transistor for different gate overdrive voltages are shown. A kink effect is observed in doped tunnel of PD devices for higher voltages. This is due to floating body effect, which is the capacitor formed between the isolated silicon tunnel and the body of transistor [20], [26].



L. Dreeskornfeld, J. Hartwich, E. Landgraf, R. J. Luyken, W. Rösner, T. Schulz, M. Städele, D. Schmitt-Landsiedel, L. Risch, "Comparison of partially and fully depleted SOI transistors down to the sub 50nm gate length regime", *Infineon Technologies-Corporate Research Otto-Hahn-Ring*, Institute for Technical Electronics, TU Munich, Germany.

Figure 4-1 Output characteristics of a 200 nm channel length device

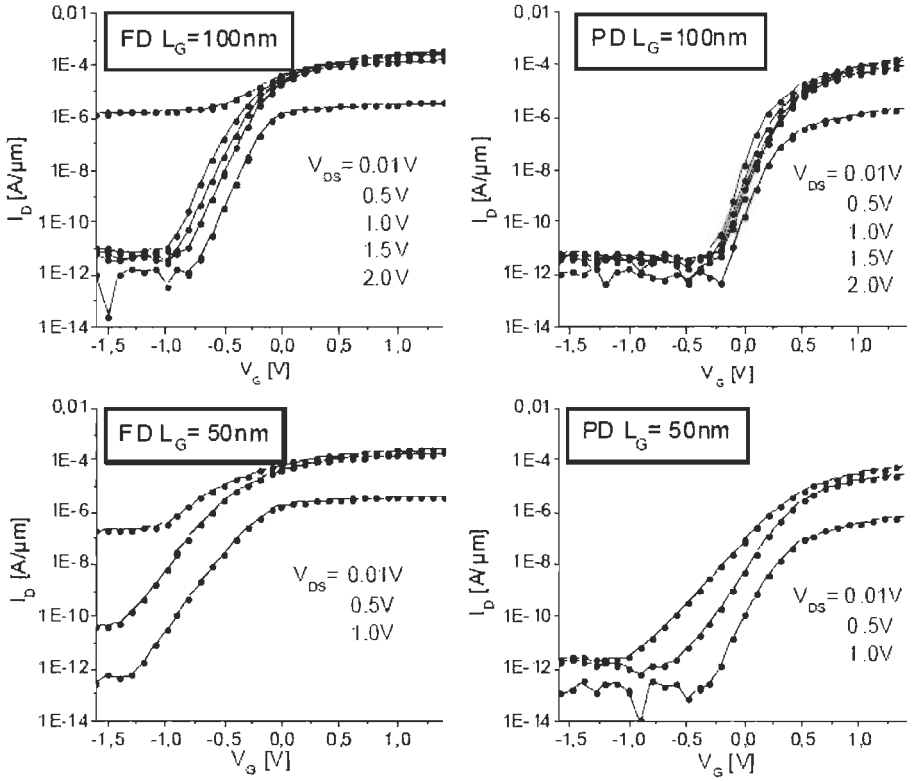
Undoped tunnel of PD device does not have this undesired effect. In addition to that it has 30 % higher on current, because of a higher mobility [26].



L. Dreeskornfeld, J. Hartwich, E. Landgraf, R. J. Luyken, W. Rösner, T. Schulz, M. Städele, D. Schmitt-Landsiedel, L. Risch, "Comparison of partially and fully depleted SOI transistors down to the sub 50nm gate length regime", *Infineon Technologies-Corporate Research Otto-Hahn-Ring*, Institute for Technical Electronics, TU Munich, Germany.

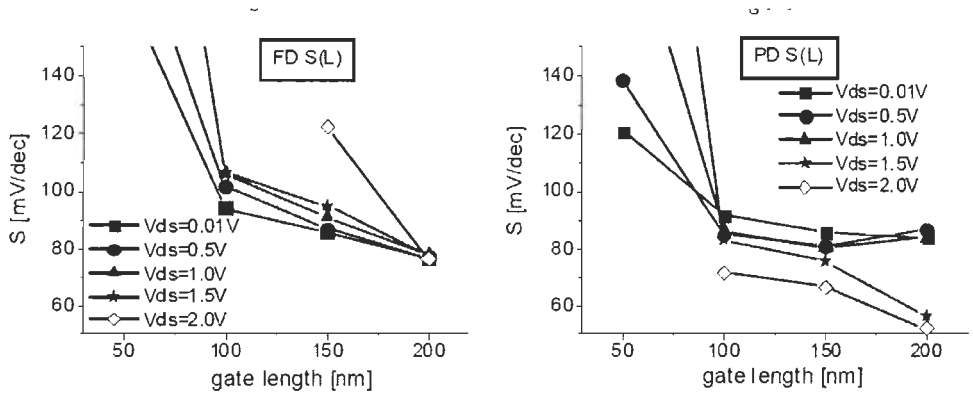
Figure 4-2 Threshold voltages

Figure 4-2 shows that the threshold voltage of a PD device is quite constant for a range of channel lengths, but FD devices have varying threshold voltages [26].



L. Dreeskornfeld, J. Hartwich, E. Landgraf, R. J. Luyken, W. Rösner, T. Schulz, M. Städele, D. Schmitt-Landsiedel, L. Risch, "Comparison of partially and fully depleted SOI transistors down to the sub 50nm gate length regime", *Infineon Technologies-Corporate Research Otto-Hahn-Ring*, Institute for Technical Electronics, TU Munich, Germany.

Figure 4-3 Transfer characteristics of 100 nm and 50 nm devices



L. Dreeskornfeld, J. Hartwich, E. Landgraf, R. J. Luyken, W. Rösner, T. Schulz, M. Städele, D. Schmitt-Landsiedel, L. Risch, "Comparison of partially and fully depleted SOI transistors down to the sub 50nm gate length regime", *Infineon Technologies-Corporate Research Otto-Hahn-Ring*, Institute for Technical Electronics, TU Munich, Germany.

Figure 4-4 Sub threshold slopes

Also as seen in transfer characteristics of 100 nm and 50 nm (Figure 4-3), PD devices have lower off current levels, which is another advantage for PD devices (especially in 50 nm channel length) [26].

Another advantage for FD devices is that as shown in Figure 4-4, they can have near ideal sub threshold slopes in small lengths [26].

It is also concluded in this article that the channel length of FD devices is limited to 4 times the thickness of silicon channel. This limit is set to guarantee the good performance of the transistor. It is generally concluded that except in some cases, FD devices have better performance than PD devices [26].

4.1.4 Modeling and simulation

Modeling and simulation tool used in this master project is nanoMOS 3.5, which is accessible through nanohub.org. This tool is originally developed by Zhibin Ren. NanoMOS tool simulates thin body fully depleted double gated n-MOSFETs using different transport models [28].

Double gate MOSFETS are devices with two gate electrodes, placed at two opposite sides of channel. This architecture gives better control of channel. Generally, these two gate

electrodes can be two surfaces of one gate or two independent electrodes controlled by two gate signals [20].

This thesis tries to solve the problem of near ballistic, non-equilibrium transport in deeply scaled transistors using non-equilibrium Green's functions. This method replaces the quasi-equilibrium, scattering transport model, valid in transistors with longer channel, and is meant to describe quantum effects that occur in nanoscale [29].

In the introduction the scaling of MOSFETs is discussed in regard to the bulk and SOI technologies. Here a resume of the discussion is presented.

When devices are scaled, a compromise between functionality and reliability is inevitable. Functionality and reliability are affected by Short Channel Effects (SCEs) [8].

Performance related examples of these effects are: In smaller channel lengths the threshold voltage rolloff causes the sub threshold swing to reduce. That makes it more difficult to turn the transistor off. Drain-induced barrier lowering (DIBL) effect causes the drain voltage to vary with threshold voltage. This is a challenge in designing circuits using these devices. Reliability examples of SCEs are gate tunneling current junction tunneling current.

Higher channel doping concentration, Thinner insulator and a higher isolator dielectric constant in relation to semiconductor dielectric constant make SCE effects less significant and make deeper scaling possible [29].

On the other hand a low channel doping concentration decreases body effect coefficient, which increases sub threshold swing. To solve this conflict, a doping profile

can be use that gives the channel a lower doping concentration near the silicon-oxide interface and a higher doping concentration elsewhere [29].

These measures and some other make it possible to scale bulk silicon transistors down to 25 nm. After that leakage currents are too high. In partially depleted SOI MOSFETs, isolation causes charge build up in channel and floating body effect. This is reduced in a fully depleted SOI device [29].

In a double gate FD MOSFET the second gate reduces the SCEs and so high doping concentration is not very much needed to fight SCE anymore. Lower doping decreases the band to band tunneling junction leakage current [29].

The introduction concludes that despite fabrication difficulties, thin body DG MOSFETs are the best choice for the continuation of scaling [29].

4.2 Geometry and fabrication

4.2.1 Fabrication Process

Double Gate SOI MOSFET is believed to be a structure with high potential for being used in deeply scaled MOSFETs. [30]

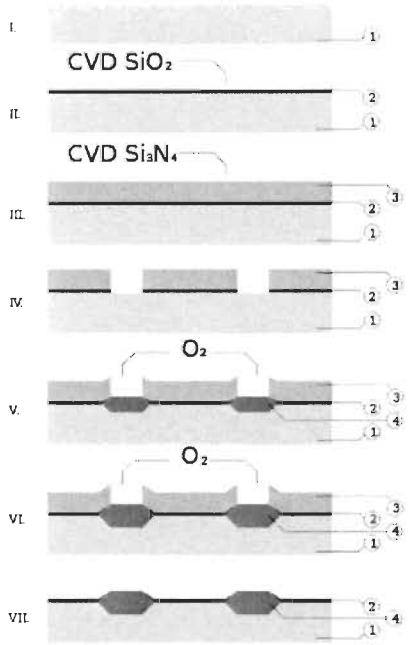
Here a fabrication process will be introduced that builds Double Gate SOI structures on wafers that are not exactly conventional SOI wafers. In this method, amorphous silicon is used as the base for making a film that behaves like an SOI film. More specifically amorphous silicon is recrystallized to form a high quality polysilicon layer that behaves similarly to single crystal silicon. This film is called large grain polysilicon on insulator or LPSOI. Double gate device that is made on this film has close characteristics as one made

on a conventional SOI film. Instead this method offers more flexibility and simplicity in fabrication process of devices. Solidworks is used to generate fabrication models and masks in this chapter [30], [33].

Bottom gate, in this method, is formed using a semi-recessed LOCOS process. It is necessary to briefly introduce this process. LOCal Oxidation of Silicon or LOCOS is a process that forms a silicon dioxide layer in a selected region on a silicon surface in a way that the silicon/oxide interface is placed beneath the surface. It was first developed to insulate different MOS transistors from each other. The steps to be taken to realize the selective oxidation on the silicon layer are as follows [20]:

First a silicon dioxide layer and then a silicon nitride layer is deposited on the silicon using Chemical Vapor Deposition (CVD) method (figure 4-5, layers 2 and 3). Then these layers are etched in selected regions so that silicon is exposed. Now if the wafer is put in thermal oxidation environment, the coating layers will prevent the oxidation of silicon in unwanted regions, hence LOCal Oxidation of Silicon. Silicon nitride permits very low diffusion of oxygen in high temperature, so it is a good coating layer for selective oxidation. [20]

When silicon is oxidized, its volume increases to 2.4 times. In a LOCOS process, this means that the coating layer is pushed upward. This expansion creates a strong tension between silicon and nitride layer that will damage the substrate. Oxide layer provides a solution: when heated during thermal oxidation, its viscosity is decreased so it can absorb the tension between two layers. [20]



www.wikipedia.org

Figure 4-5 Semi recessed LOCOS process

When the oxide is grown to the desired amount, the oxidation is stopped and the nitride mask is removed. What was explained is semi-recessed LOCOS, in which oxidation starts from the actual level of silicon layer. In fully recessed LOCOS before starting thermal oxidation, a little silicon is etched so that oxidation starts from a lower level, which creates a more deeply buried local oxide. [20]

In the following process, LOCOS method is employed to form the bottom gate. First bottom gate area is selected and coated with an oxide and a nitride layer (Figure 4-6) and then Thermal oxidation is performed. Figure 4-7 shows the mask used to etch oxide and nitride layers and leave the selected areas of silicon exposed to oxidation. This LOCOS process creates bottom gate edges with more planar-like features. It means that the process

softens the sharp edges of bottom gate and that decreases the reliability problems caused by these edges (Figure 4-8). Then by Arsenic (a substance from group V) implant the bottom gate is doped. A high quality bottom gate oxide layer, with 270 Angstrom thickness, is also formed by thermal oxidation. [30]

3D models are all created using solidworks [33].

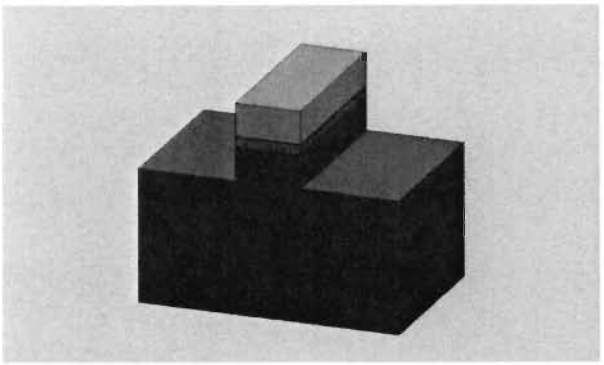


Figure 4-6 Preparation for LOCOS process

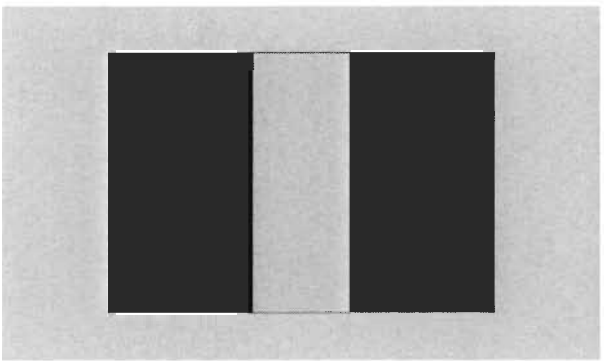


Figure 4-7 Mask used in LOCOS process to etch oxide and nitride layers

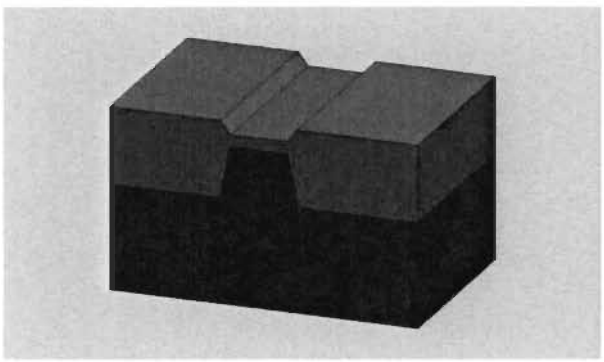


Figure 4-8 Finished LOCOS process with bottom gate area to be implanted (red) and oxide (blue)

Now that bottom gate and its oxide layer are formed, to make the body of the transistor, 500 to 1000 Angstrom of amorphous silicon is deposited on the gate oxide to be

crystallized into an LPSOI layer. This is done through Metal Induced Lateral Crystallization (MILC), which is itself a form of Metal Induced Crystallization (MIC). MIC is a method of transforming amorphous silicon to polycrystalline silicon in relatively low temperature using a metal as the seed for crystallization. An annealing step in temperatures between 150 and 400 is also included to let the crystals grow. In MILC method, instead of coating the amorphous silicon with the seed metal, metal is put in contact with silicon only in one small area and then the crystal is grown gradually and laterally, starting from the metal seed and expanding throughout the amorphous silicon. In this way metal contamination is much lower because the silicon/metal contact is much more limited. In transistor applications, to grow a polycrystalline channel from amorphous silicon, metal can be deposited in source or drain region to better keep the channel from metal contamination [20], [31].

Here the amorphous silicon (deposited in a Low Pressure Chemical Vapor Deposition, LPCVD process using SiH4 as Silicon source) is transformed in a Nickel Induced Crystallization process in 580 C for 24 hours [30], [31].

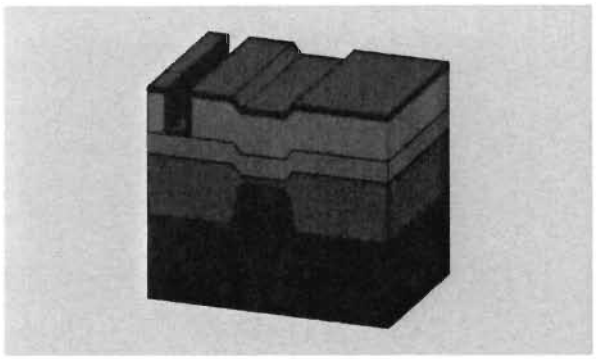


Figure 4-9 Amorphous silicon deposited on top of oxide layer, also LPO layer (green) and Nickel (violet) are deposited to start crystallization.

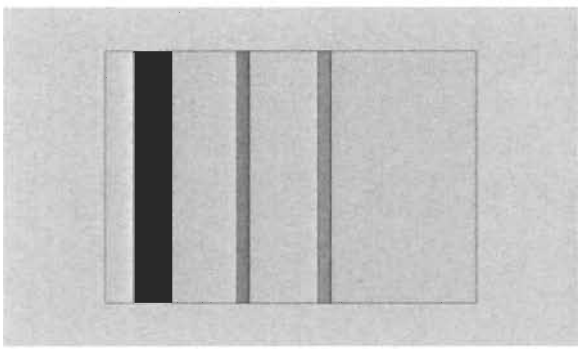


Figure 4-10 Mask used to etch the LPO

To create the seeding window for Nickel deposition, 3000 Angstrom of Low Temperature Oxide (LPO) is deposited and etched in the side. Then 100 Angstrom of Ni is deposited and lateral crystallization is started (Figure 4-9). Figure 4-10 shows the mask used to etch the LPO to make the contact window for Nickel. When crystallization is finished after several hours, the Ni is removed with H2SO4:H2O2 (4:1) in 70 C and then the sacrificial oxide layer is also removed with HF. Then the annealing process is performed at temperatures less than 850 C which increases the grain sizes and completes the process [30], [31].

Then the channel is implanted with B and then 100 Angstrom oxide is grown as the top gate oxide layer (Figure 4-11). Figure 4-12 shows the mask used to etch the crystallized polysilicon of channel. Then a poly-crystalline layer is deposited on the oxide layer (Figure 4-13). This poly layer is then etched to form the top gate (Figure 4-14) and at last the top gate and the source and drain regions are implanted with arsenic ions. [30]

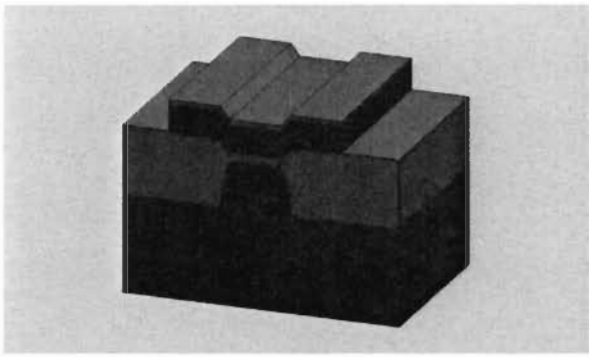


Figure 4-11 Crystallized channel, ready to be implanted with B ions

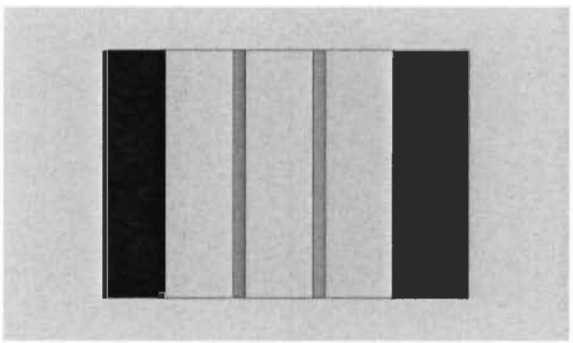


Figure 4-12 Mask used to etch the channel silicon

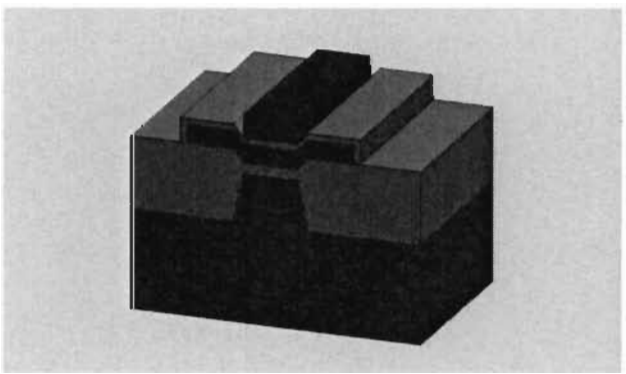


Figure 4-13 Deposition of upper gate polysilicon layer

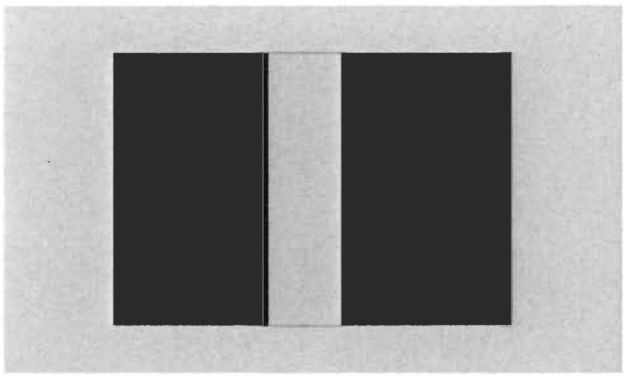


Figure 4-14 Mask used to form the upper gate

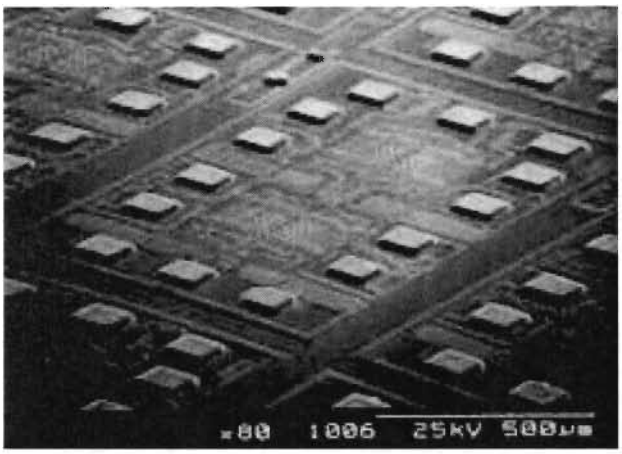
4.2.2 Packaging

A processor IC as the main way semiconductor industry sells MOSFETs has been chosen as the subject of packaging study. There are two major steps in packaging of a processor: attaching the die to the package substrate and connecting the package substrate to the PCB [32].

Wire bonding is the primary and still most frequently used method of connecting die attachment. It consists of simply stretching a golden, copper or aluminum wire between the pads on the die and the connectors. Wire bonding can be used to connect dies to other dies or PCBs to other PCBs too. This technology has been developing considerably, but today and in the case of processors, where the frequency, density and power dissipation is high, wire bonding has been replaced by flip chip technology [20], [23].

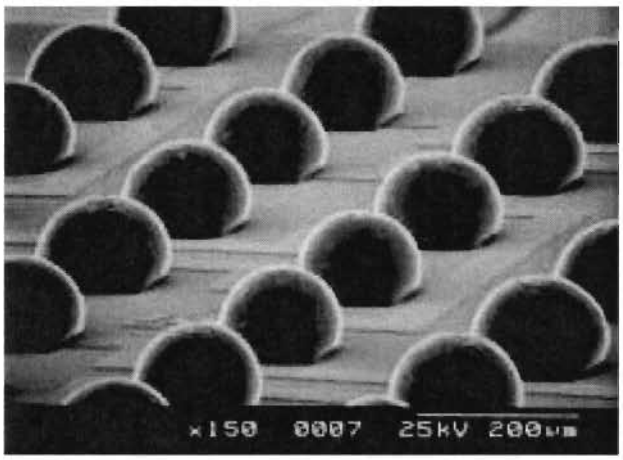
No wire is used in flip chip technology, instead the die, with its pads made on its top side, is flipped and put directly on the board (this is why it is also called Direct Chip Attach or DCA). Flip chip reduces the occupied area and height of a chip significantly. So it is

ideal for applications where space is limited. Also because bond wires are eliminated, inductance and capacitance of the connection are reduced up to 10 times and connection path is shortened 25 to 100 times. This gives it a much better electrical performance and suitable for high speed applications. In regard to I/O capability, flip chip offers a big advantage: in wire bonding, putting pads far from perimeter of the die makes the die attachment very difficult and so the inside area of the die is never used. In flip chip technology all the surface of the die is used for connections. This increases the number of pads possible on a die, a much needed feature by increasingly dense processors. Mechanically, flip chip connections with their adhesive underfill in place, are the strongest connection type. At last, in high volume flip chip can be the least costly connection type [32].



www.flipchips.com

Figure 4-15 Electroless Ni-Au UBM



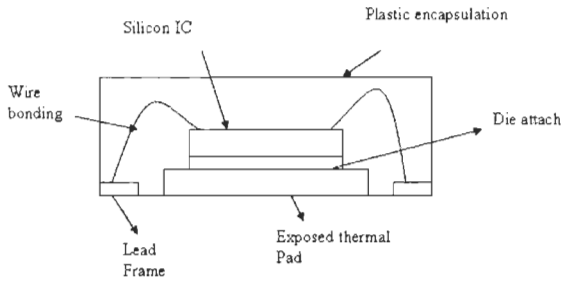
www.flipchips.com

Figure 4-16 Reflowed solder bumps on electroless Ni-Au UBM

Solder bump flip chip is the most common type of flip chips. There are four steps in a solder bump flip chip process: First step is preparing pads to be connected to solder. The second step is to put the bumpers on the die pads. These bumpers are the bulging parts that make the connection. Then the die is aligned and attached to the substrate using a reflow

process (technically means deforming plastically). At last the space under the die is "under filled" by an insulate adhesive. In addition to electrical connection, bumps provide a thermal path for heat dissipation from die to substrate. "Sputtered UBM/printed solder" is the chosen solder bump flip chip process. It has low cost, good composition control, compatibility with different alloys (lead-free, non-binary...) and acceptable precision (150 um) [32].

"Micro lead frame" or "flat no lead" technology is a near "Chip Scale Package" (a package whose area is not more than 1.2 times the area of the die) package to substrate connection method which uses perimeter leads or pins (tens to two or three hundreds of pins in one or two rows) to connect to the board. Die is attached by wire bonding to the leads on the frame or perimeter of the package. The die is in contact with a thermal pad which can be on top or on the bottom of the package. Package is connected to the board by soldering using a solder paste. This is a low cost, reliable and thermally effective package [20], [23].



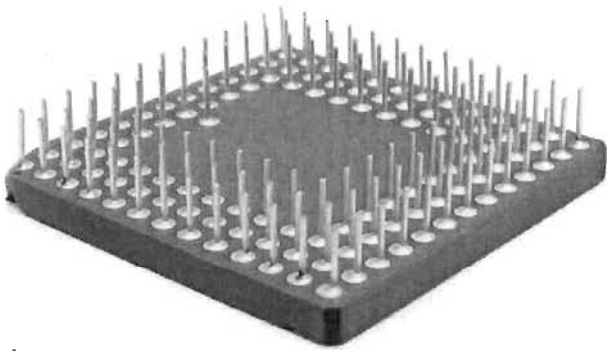
Section View of a QFN package

www.wikipedia.org

Figure 4-17 A "Quad Flat No lead" structure

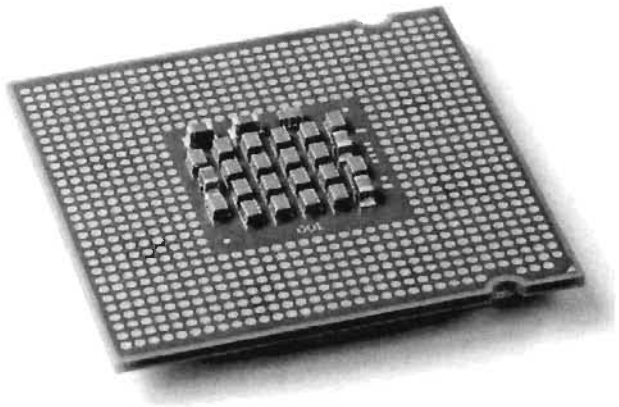
"Grid array" packages are the technology to be employed when high I/O count chips with highest technical need are to be used. In this technology the connectors at the bottom of the package are patterned in a square array that might cover the entire bottom surface of the package. Pin Grid Array (PGA) is a kind of grid array package in which connectors are pins coming out of package bottom surface. These pins can connect to the board with through holes or using a socket. From 1990 PGA has been the dominant package used for Intel and AMD processors. This continued almost until 2004 (Intel Pentium 4 and AMD Athlon 64) when Land Grid Array (LGA) appeared. In LGA the pins are moved to the socket. Instead on the package bottom surface, where pins would have been, there is an array of pads that make the contact with pins on socket. Pads are made of Cu covered with a layer of Au. Latest processors need more than 1000 pins and LGA provides a higher pin density, compared to PGA. This technology is currently the dominant package solution in processor industry. In "Ball Grid Array" (BGA) pins or pads, discussed above, are replaced with solder bumps. The package can be mounted on a PCB directly, without solder paste

and with a reflow process, with the least height and resulting in a very good connection. Fine pitch BGA (FBGA) is a branch of BGA with the highest density of contacts among package to substrate connection types. In 2014 it will reach to 100 um array pitch. BGA packages have higher thermal conductivity between the package and the board and lower inductance, due to direct contacts [20], [23].



www.wikipedia.org

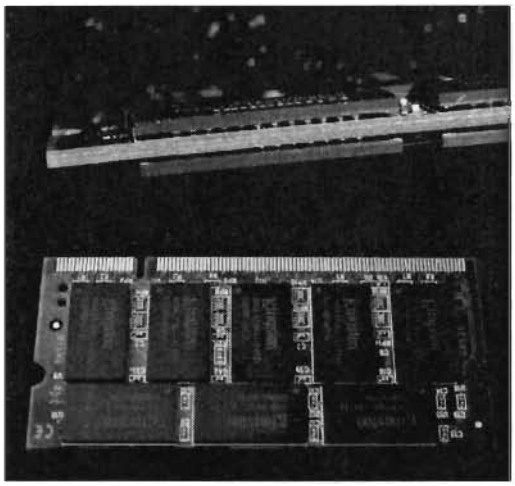
Figure 4-18 Pin Grid Array at the bottom of a processor



www.wikipedia.org

Figure 4-19 An LGA processor

BGA has technical advantage over LGA. It has more bump density, smaller size and less parasitic effects and better thermal conductivity. But it needs to be connected to the board by a machine and cannot be removed and replaced. Today, big producers use LGA technology with a socket on the mother board for PC applications where removable processors are needed. BGA instead can be used when consumer's choice is limited and size and mechanical uniformity of the device is important (very low cost laptops and notebooks, for children or in laptop distribution programs in developing countries). In smaller processing units used in household devices, cars, watches and other electronic devices, BGA might have a wider application.



www.wikipedia.org

Figure 4-20 BGA RAM ICs connected to a PCB

Today, one important trend in material development is to make it green. In wire bonding, flip chip and die attachment techniques, lead free regulations have been accepted in many countries. Underfill materials, thermal interface materials and package substrates move toward halogen-free materials [23].

Substrate is the most expensive part of packages. It also imposes parasitic effects and performance limitations more than any other part of packaging. Smaller feature size, less thermal expansion and better electrical performance are the desired characteristics in substrates. To increase speed of circuits without parasitic capacitance effects from package, low k materials are used in substrate. Now materials with dielectric constant of 3.4 are usable in industry the ones with dielectric constant of 2.8 is in research. Moving toward halogen-free materials has challenged lowering dielectric constant [23].

4.3 Numerical modeling

NanoMOS 2.0 is a simulation tool for ballistic and non-equilibrium transport in deeply scaled MOSFETs. It is accessible on nanohub.org and employs three transport models with different levels of quantum effects consideration to simulate Thin Body Fully Depleted Double Gated n-MOSFETs [28].

After a brief introduction, some simple NanoMOS' simulation results are shown. Then NanoHUB website and NanoMOS as a tool will be explained in more details.

NanoHUB is a nanotechnology web-based resource for educational and academic purposes. It includes online courses and educational documents and learning modules to help students and researchers become familiar with nanotechnology in a general or in a technical and academic level. It also includes a large number of simulators that simulate various phenomena and devices in nanoscale. These simulators vary in complexity and goal: some are designed for general education about a certain aspect of nanoscience and some perform detailed simulations on a specific device. NanoHUB benefits from the contribution of more than 600 researchers and educators and the number of its users and simulations done using its tools is rapidly growing [34].

One can sign in to nanoHUB as a researcher and access the tools including NanoMOS directly. In this way a communication window is opened between user's web browser and the tool through nanoHUB and input is given by the user to the system step by step and the program will run on a remote computer and the downloadable results are presented to the user. In many cases tools are open source and downloadable as it is the case for NanoMOS presently. In this way users can run the code on their own computer. In this project, of

course, all of the measurements and manipulations have been done on open source MATLAB code of NanoMOS [29], [34].

In NanoMOS, User can choose to simulate a non-equilibrium thin-body transistor and calculate drain current-drain to source voltage and drain current-gate to source voltage plots. The selected device can be simulated using drift diffusion transport, semi classical ballistic transport and ballistic transport with Green's function approach. Each one of these models takes into account a curtain amount of quantum phenomena and produces the results with curtain exactitude. Naturally each one consumes a curtain amount of computation resources. Then user can choose a bias for their simulation including both gates and source and drain voltages and the signal step sizes. Then device geometry, doping profiles, wafer material and insulators dielectric constants are defined. At the end some simulation options (equations, convergence parameters and other technical options) are chosen [28].

User can choose among a wide range of results. In accordance to the importance of quantum aspect in these devices, most of them are related to quantum parameters (electron density profiles, energy profiles...), but it is also possible to see some simple electronic plots like the ones discussed above [28].

Using this tool, drain current variations of thin-body fully-depleted double-gate MOSFET has been studied in relation to changes in different parameters. The goal is to observe and measure the effects of different parameters, discussed in Physical and Mathematical modeling sub-chapter, on MOSFET's performance. Table 4-2 contains device parameters and Table 4-3 bias parameters [28].

Table 4-2 Simulation device parameters

parameter	Quantity
Temperature	300
Source/drain doping concentration (/cm3)	1e+20
Source/drain Gaussian doping profile slope	0
Material and wafer orientation	Si, wafer(001), trans(100)
Top/bottom insulator relative dielectric constant	3.9
Body relative dielectric constant	11.7
Silicon film thickness (nm)	1.5
Gate isolator thickness (nm)	1.5

Table 4-3 Bias parameters

parameter	Quantity
Top/Bottom gate voltage (V)	0.6
Source voltage (V)	0
Drain voltage (V)	0 : 0.05 : 1.4

In Figure 4-21, drain current is shown for gate lengths of 3, 5, 10 and 20 nm. A respective decline in current is observed that accounts for increase in canal resistance.

By decreasing the thickness of gate isolators from 1.5 nm to 1 nm, the maximum drain current is increased from 1.11e+3 to 1.21e+3 ($\mu A / \mu m$). (Figure 4-22)

Increasing the relative dielectric constant of gate isolators from 3.9 to 6, more charges are absorbed to canal and current increases. (Figure 4-23)

Finally by employing a doping profile in the channel, the performance of channel is improved and current increases. (Figure 4-24)

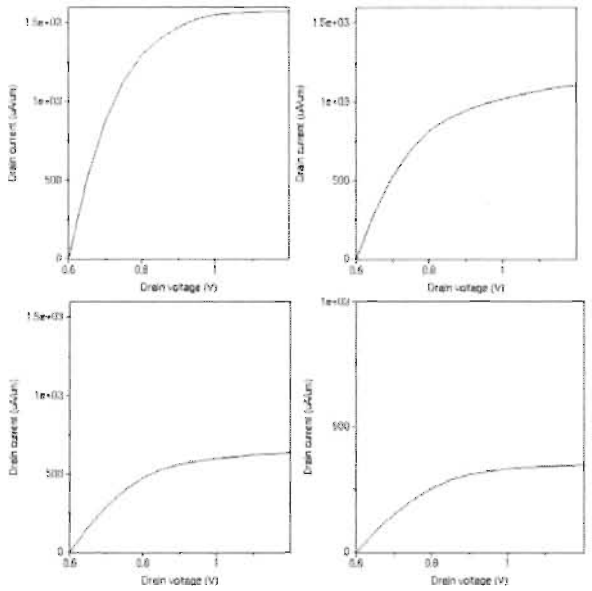


Figure 4-21 Drain current for different gate lengths

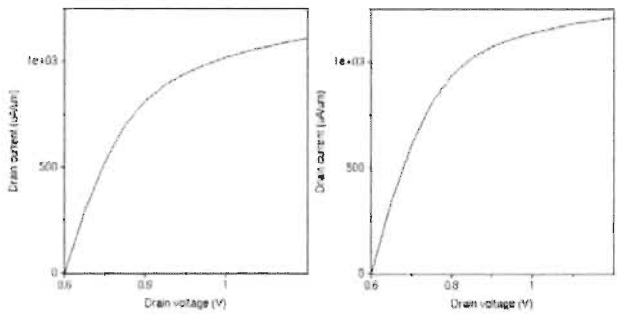


Figure 4-22 Drain current for different gate isolator thicknesses

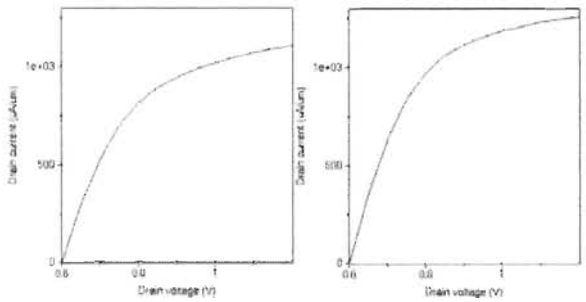


Figure 4-23 Drain current for different gate isolator dielectric constants

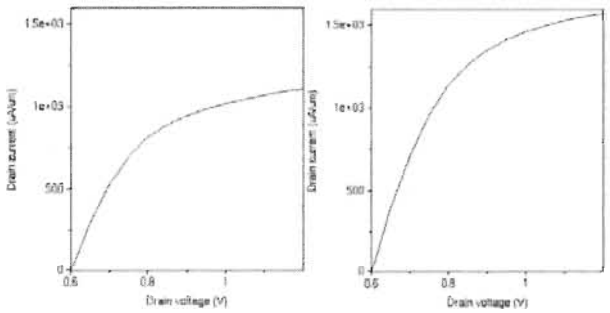


Figure 4-24 Drain current for different Gaussian doping profile slopes

NanoMOS gets its input as an input file that includes different input data categories. The first thing that input file defines is device structure. Sizes and positions are defined with symmetry in the direction of x and with y coordination starting from top of the top oxide layer. As it is explained in more detailed in the next chapter, a grid is defined on the device's plan that is used in solving the model with matrix calculation method. The size of grid elements needs to be given as input too. These are device and grid input parameters as explained by [35]:

nsd: Source/Drain doping concentration ($1/cm^3$)

nbody: Body doping concentration ($1/cm^3$)

lgtop: Length of the top gate (nm)

lgbot: Length of the bottom gate (nm)

lsd: Length of the Source/Drain (nm)

overlap_s: Source extension length (nm)

overlap_d: Drain extension length (nm)

dopslope_s: Source Gaussian doping profile slope (dec/nm)

dopslope_d: Drain Gaussian doping profile slope (dec/nm)

tox_top: Top insulator thickness (nm)

tox_bot: Bottom insulator thickness (nm)

tsi: Silicon film thickness (nm)

temp: Lattice temperature (K)

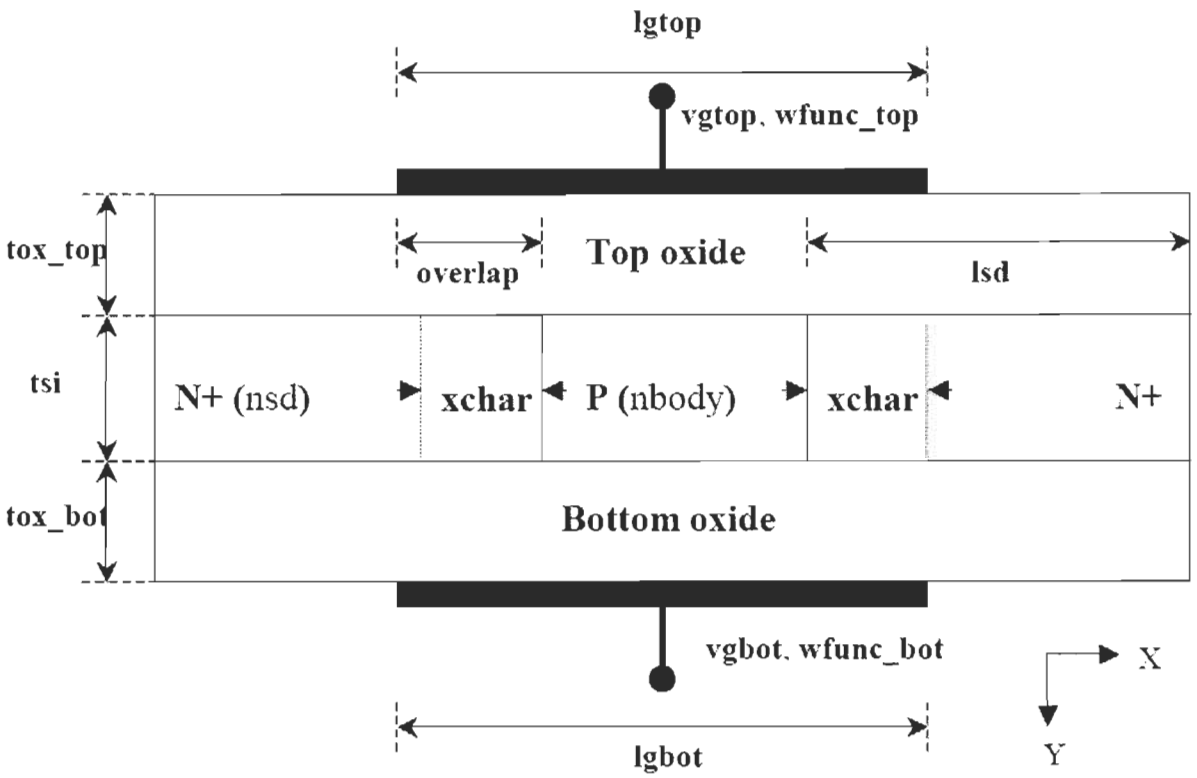
dx: Horizontal node spacing (nm)

dy: Vertical node spacing (nm)

refine: Refining factor for vertical grid (used in schred.m)

A positive value for overlap means that gate extends over drain or source. A negative value refers to an underlap: gate does not reach drain or source in x direction. A Gaussian profile is assumed for doping profile in source and drain. "dopslope" parameters define the slope of this Gaussian profile. If they are set to zero, the doping profile is a step function. In solving Schrödinger equation (in schred function) a refined grid is used. That is where

refine parameter is used. Bellow a plan of device and device parameters is shown [29], [35].



ZhibinRen, with contributions by: Ramesh Venugopal, Jung-HoonRhew, Jing Guo, Dave Rumsey, Dr. SebastienGoasguen, Prof. SupriyoDatta and Prof. Mark S. Lundstrom, "NanoMOS 2.0 A 2D-Simulator for Double-gate MOSFETs", *Manual*, Purdue University, Decembre 2001.

Figure 4-25 Device parameters

Five transport models are available for simulations. In Classical Ballistic Transport model (CLBTE) and Quantum ballistic Transport model (QBTE) in each point of x (channel direction) a 1D Schrödinger equation is solved and subband profiles of electrons are found (ESUB(x)). Then in CLBTE carrier transport in each subband is found using thermionic emission. Thermionic emission is the flow of carriers over a potential barrier due to heat. In this model the tunneling through the barrier between source and

semiconductor is not considered. In QBTE transport of carriers in x direction is calculated by solving a 1D Schrödinger equation in x direction using Non-Equilibrium Green's Function method (NEGF). Source-channel barrier quantum tunneling is considered in this model. Drift diffusion (DD) is a version of classical drift diffusion model with quantum corrections. Subband profiles in y direction are found, as in two previous cases, quantum mechanically by solving 1D Schrödinger equation. To find the carrier transport in each subband a 1D drift diffusion equation is written in x direction. Mobility is found with Caughey-Tomas model (equ. 4.8) where *E is electrical field in x direction and other parameters are given by user [29], [35].

$$\mu(E) = \frac{mu\_low}{[1 + (\frac{E \cdot mu\_low}{vel\_sat})^{beta}]^{1/beta}} \quad (4.8)$$

Energy transport model (et) is a 1D energy transport model that works by calculating thermal flow in a gas of electrons moving in3D [29], [35].

Quantum Dissipative Transport model (QDTE) calculates ESUB(x) or subband profiles quantum mechanically as in CLBTE. Then it employs Green's function method to find a dissipative model of transport, meaning that it accounts for scattering of carriers to describe the 2D transport [29], [35].

Bellow a list of transport model related parameters that input file must contain is presented as explained in [26]. The last two are related to energy transport model [35].

model: transport Model (DD, CLBTE,QBTE,ET,QDTE)

mu_low: Low field mobility (cm/s)

beta: Caughy-Thomas parameter (dimensionless)

vsat: Saturation velocity (cm/s).

ELE_TAUW: Electron energy relaxation time (s)

ELE_CQ: Energy flux related parameter (dimensionless)

These are bias parameters as explained in [35]:

vgtop: Top gate voltage

vgbot: Bottom gate voltage

vs: Source contact voltage

vd: Drain contact voltage.

vd_initial: Drain start voltage.

vgstep: Step size for the gate voltage

ngstep: Number of gate voltage steps

vdstep: Step size for the drain voltage

ndstep: Number of drain voltage steps

vd_init is the initial guess in iterative process of solving the equations for drain voltage. It is recommended that to increase the convergence rate of iterationsvd_init should be set bellow vd value. If ndstep is nonzero, drain current will be plotted versus drain voltage and if it is zero, drain current will be plotted versus gate voltage [35].

These are material parameters (mostly of quantum nature) as explained by [35]:

wfunc_top: Top gate contact work function (eV)

wfunc_bot: Bottom gate contact work function (eV)

mlong: Longitudinal relative electron mass ratio

mtrans: Transverse relative electron mass ratio

eps_top: Top insulator relative dielectric constant

kox_top: Top insulator relative dielectric constant

kox_bot: Bottom insulator relative dielectric constant

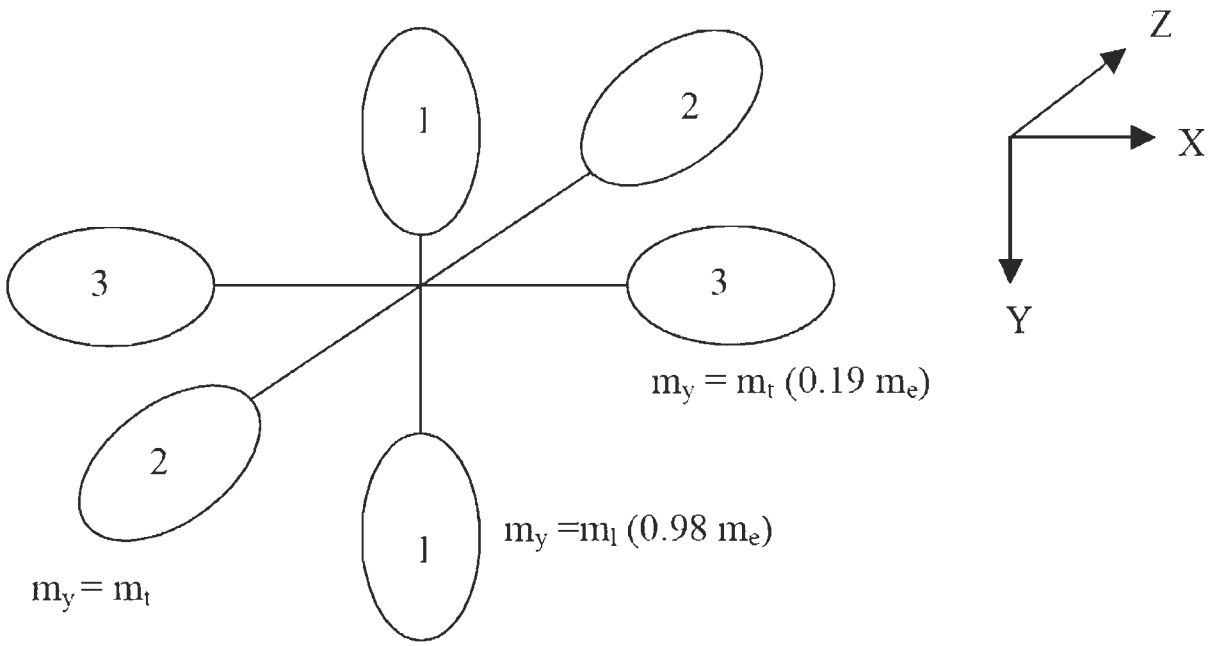
dec_top: Conduction band offset between the substrate and the top gate insulator (eV)

dec_bot: Conduction band offset between the substrate and the bottom gate insulator (eV)

Self-consistent solution of equations in NanoMOS is dependent on two divergence parameters. One is "dvpois" which sets the minimum difference needed between potential values resulted from two iterations of solving Poisson equation. If the difference between potentials is less than dvpois then the solution has converged to the final result. Other one is "dvmax" which does the same for the potentials resulted from self-consistent solution of Poisson equation and Schrödinger equation [29], [35].

There are some options to be set in input file too. "ox_penetrate" is a flag that determines if, when solving 1D Schrödinger equation in y direction, the electron penetration in the oxide layer should be considered or not. In chapter 5 it is explained in more details how this is done. "dg" is another flag that couples the ramping of top and bottom gate voltages, if it is set to true. If it's false, the bottom gate will stay at "vgbot" value and does not change. When using drift diffusion model, user can choose to apply

Fermi-Dirac or Maxwell-Boltzmann statistics in solving the transport model. When using other models, Fermi-Dirac statistics is automatically applied. Electrons in different valleys have different effective masses and it influences their mobility (Figure 4-26). Valleys in y direction impose much more effective masses and are called unprimed valleys. "valleys" flag determines if only electrons in unprimed valleys should be considered in simulation or electrons in all valleys. "num_subbands" tells the number of subbands in each valley that must be considered in solving 1D Schrödinger equation. Usually only lowest subbands are occupied by electrons [29], [35].



ZhibinRen, with contributions by: Ramesh Venugopal, Jung-HoonRhew, Jing Guo, Dave Rumsey, Dr. SebastienGoasguen, Prof. SupriyoDatta and Prof. Mark S. Lundstrom, "NanoMOS 2.0 A 2D-Simulator for Double-gate MOSFETs", *Manual*, Purdue University, Decembre 2001.

Figure 4-26 Valleys and electron masses

User can choose the plots to be saved and showed using following flags as explained in [35]:

I_V: Drain current versus drain voltage characteristics.

Ec3d: Three-dimensional plot of the conduction band.

Ne3d: Three-dimensional plot of the carrier concentration band.

Ec_sub: Energy profile of the different subbands along the channel.

Ne_sub: Carrier concentration of the different subbands along the channel.

Te: Carrier temperature along the channel (eV).

Ec_IV: Energy profile of the lowest subbands.

Ne_IV: Carrier concentration of the lowest subbands.

The I_V characteristics will be plotted versus gate voltage Ifndstep is set to zero. If more than one bias point has been simulated, all other plots are generated for the last bias point that has been simulated [35].

At the end of each simulation, data like convergence history, I-V characteristics, subband energies, electron densities, potential profile and carrier temperature are saved to files with .dat and .psextensions [29], [35].

Chapitre 5 - Implementation

5.1 Mathematical formulation

As it has been stated qualitatively in previous chapters, conventional theories of carrier transport cannot predict the behavior of MOSFET channels in nanoscale. In this chapter we see a more detailed description of transport models that describe these devices more accurately.

Conventional models of carrier transport are derived from Boltzmann Transport Equation (BTE). These models are formed on a scattering based view of carrier transport. This, being the case for long channel devices, fails to models quasi-ballistic transport that occurs in very short channel devices. [29]

BTE deals with statistical distribution of particles in the matter. It is applied when a system is far from thermodynamic equilibrium, for example when there is an electric field applied to a region (which is the case with MOSFETs) or when a temperature gradient is applied. In these cases BTE can describe how current flows (how charges move) or how heat flows (movement of heat carriers) [30].

BTE describes statistically the position and momentum of particles as a function of time. $f(r,p,t)$ is a density function so that $f(r,p,t)drdp$ is the number of particles in a neighborhood of r and with a momentum within a neighborhood of p in time t. Now if an

external force F is applied to particles with mass m, without considering any collisions, we will have [20], [29]:

$$f(r+\frac{p}{m}dt, p+Fdt, t+dt)drdp = f(r,p,t)drdp \quad (5\text{-}1)$$

Meaning that after time dt has passed, particles that were in r with momentum p, would have moved to $x+\frac{p}{m}dt$ and their momentum would have changed to $p+fdt$. Now if we add the effect of collisions of particles as $\frac{\rho f}{\rho t}|coll$ (founded by quantum calculations), we will have [20], [29]:

$$\frac{\partial f}{\partial t}+\frac{\partial f}{\partial r}.\frac{p}{m}+\frac{\partial f}{\partial p}.F=\frac{\partial f}{\partial t}|coll \quad (5\text{-}2)$$

Solving BTE is a hugely complicated problem and still it fails to capture any non-classical carrier transport features. To do so, Non-Equilibrium Green's Function (NEGF) formalism is a very appropriate method to calculate nanoscale effects. Green's functions, in mathematics, are functions used to solve inhomogeneous differential equations with some boundary conditions. In Physics this name is often used more generally to refer to many correlation functions [20], [29].

In NEGF approach, electrons and phonons, which are carriers of current and heat, create what is called quantum fields. Previously it was mentioned that these carriers do not have equilibrium distribution in nanoscale. NEGF is a method of solving the non-equilibrium dynamic equation that describes non-equilibrium distribution of these particles. A field operator, $\Psi(r)$ (r being space-time, written as $r=(\vec{r},t)$) is defined in relation to quantum fields of these particles. Then Green's functions are written in as functions of

these field operators: $<\Psi^{+}(r_2)\Psi(r_1)>$ or $<\Psi(r_1)\Psi^{+}(r_2)>$. Brackets average field operators' product over states of the system. So the functions define the relation between a field operator in space-time r_1 and its conjugate in space-time r_2. In this way Green's functions describe the correlation of carriers in r_1 and r_2 [29].

Fourier transform is used to transform these correlation functions from time domain to energy domain and finally two equations that describe non-equilibrium transport can be derived as [29]:

$$G(E) = [EI - H_0(E) - \Sigma(E)]^{-1} \quad (5\text{-}3)$$

$$G^n(E) = G(E)\Sigma^{in}(E)G^{+}(E), G^p(E) = G(E)\Sigma^{out}(E)G^{+}(E) \quad (5\text{-}4)$$

G is retarded Green's function and G^{+} is its Hermitian conjugate or the advanced Green's function. G^n and G^p are correlation functions that describe electron and hole density. Σ, Σ^{in} and Σ^{out} are self-energy functions. H is the single-electron effective mass Hamiltonian, related to band structure and Hartree potential of electron-electron interactions. This scalar potential is found by solving the coupled Poisson equation. Solving these correlation functions self-consistently with Poisson equation will produce electron and current density spectrum equations [29]:

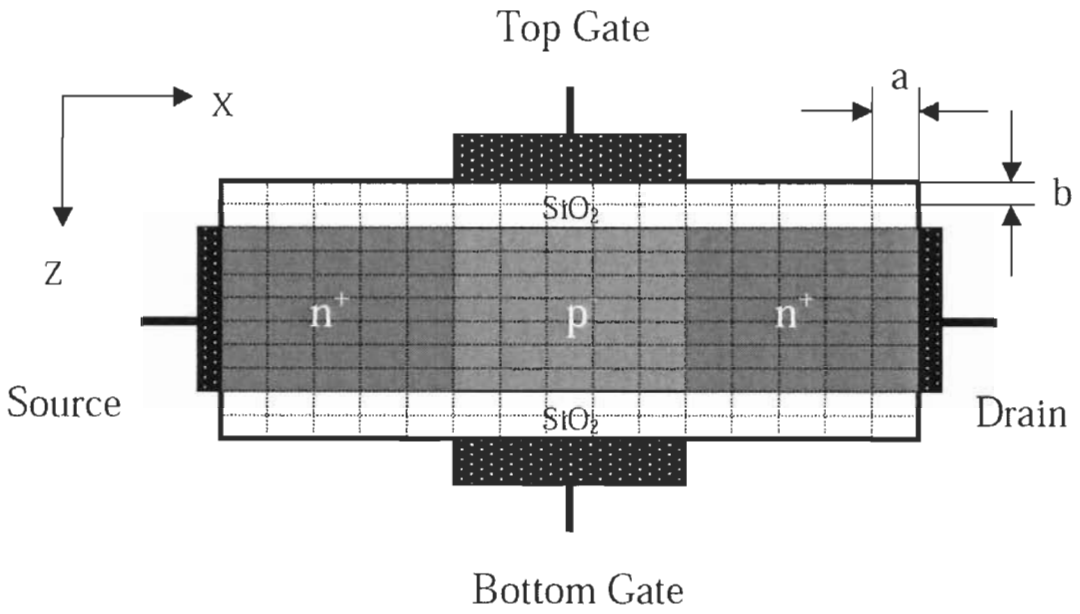
$$n(E,m) = \frac{1}{2\pi} G_m^n, \quad (5\text{-}5)$$

$$I_m(E) = \frac{q}{h} trace[\Sigma_m^{in} G^p - \Sigma_m^{out} G_n] \quad (5\text{-}6)$$

Where $n(E,m)$ is the electron density spectrum and m is a discretized unit cell. G_m^n is the mth diagonal entry of G^n and $I_m(E)$ is the current spectrum at terminal m. Σ_m^{in} is the mth diagonal entry of Σ^{in}. q is the elementary charge constant and h is the Plank constant [29].

To widen our view over this approach we see how this approach is implemented to a study of a MOSFET with ballistic transport. In such a simulation, reservoirs of carrier, which are source and drain, are considered in equilibrium. Their potential difference is represented as their different Fermi energy levels. The interaction between source and drain and the active region (channel), which constitutes the transport carrier system is represented in Σ, Σ^{in} and Σ^{out}. Interactions of electrons with each other are represented in H as mentioned above. With no other interactions, the transport is ballistic. Σ^{in} and Σ^{out} can be written in terms of Σ and Fermi energy levels. Having the Poisson equation to solve for H, Σ can be solved too [29].

When simulating short channel MOSFETs, it is important to model short channel effects and ballistic transport in two dimensions. Here we will review the solution of Poisson equation and Schrödinger equation in 2D. NEGF technique is used to solve Schrödinger equation. This 2D solution can be very computationally expensive. In the case of ultra-scaled SOI MOSFET equations can be simplified using mode-space technique to a great extent. Poisson equation will serve to produce potential profile and transport equation produces charge and current densities. Modeled device is shown in Figure 5-1. The grid elements in the Figure 5-1 will serve as elements of matrix calculation [29].



ZhibinRen, "NANOSCALE MOSFETS: PHYSICS, SIMULATION AND DESIGN", thesis, Purdue University, Octobre 2001.

Figure 5-1 Device grid

To solve the Poisson equation, Gauss's law is used:

$$\oiint[\varepsilon\bar{E}(x,z)].d\bar{S} = \int_{\Omega} q[p - n + N_D - N_A]d\Omega \tag{5-7}$$

$\bar{E}$ is electric field, p and n are hole and electron densities and N_D and N_A are donor and acceptor's doping concentrations, q is elementary electric charge and ε is dielectric constant of the materials. If we suppose that grid node numbers in x and z directions are N_x and N_z, solving Poisson will produce $N_x \times N_z$ potential values (one for each node). To find these unknowns $N_x \times N_z$ equations must be used, for which we can write and solve Poisson in all of the internal nodes or we can use boundary conditions in all of the boundary nodes. For the internal nodes, using central difference approximation (field on a mesh boundary is

equal to the average of fields in central nodes of neighboring meshes) and replacing electrical field with potential, (5-7) can be written as (5-8) for a node *mn*. *a* and *b* are taken from Figure 5.1. By assigning smaller value for b the grid will be finer in *z* direction and a more precise result for the ultra-thin channel is possible. We suppose that *p* is zero in fully depleted ultra-thin body nMOSFET [29].

$$\frac{a}{b}V_{m-1.n}+\frac{b}{a}V_{m.n-1}-2(\frac{a}{b}+\frac{b}{a})V_{m,n}+\frac{b}{a}V_{m.n+1}+\frac{a}{b}V_{m+1.n}=-\frac{ab}{\varepsilon}q(N_D-N_A-n)_{m.n} \qquad (5\text{-}8)$$

ε is determined according to the material of the region in which the mesh is placed. When a mesh is on the boundary of Si and oxide (with ε_{Top} and ε_{Bot} for the material on the top and on the bottom) layers (5-8) becomes [29]:

$$\frac{a}{b}V_{m-1.n}+\frac{b}{2a}(1+\frac{\varepsilon_{Bot}}{\varepsilon_{Top}})V_{m.n-1}-(\frac{a}{b}+\frac{b}{a})(1+\frac{\varepsilon_{Bot}}{\varepsilon_{Top}})V_{m,n}+\frac{b}{2a}(1+\frac{\varepsilon_{Bot}}{\varepsilon_{Top}})V_{m.n+1}+\frac{a}{b}\frac{\varepsilon_{Bot}}{\varepsilon_{Top}}V_{m+1.n}$$
$$=-\frac{ab}{\varepsilon_{Top}}q(N_D-N_A-n)_{m.n}$$

(5-9)

In boundary nodes the same can be done except for the nodes neighbor with gate, source or drain electrodes. In gate contacts Dirichlet boundary condition is valid which determines the node potential in a gate region node equal to vacuum potential of the gate [29].

$$V_{m,n}=V_G \qquad (5\text{-}10)$$

At source/drain contacts, Neumann boundary conditions impose that the net charge is zero [29]:

$$\vec{n}.\vec{\nabla}V=0 \qquad (5\text{-}11)$$

It means that potential is floating and will be determined in a way that net charge is zero in the contact. If a more fixed condition for potential is chosen, ballistic transport will not be properly modeled, especially that ballistic effects are stronger in these regions. For other boundary nodes we just need to suppose that no electric field exits the channel, meaning that the potential on the outer mesh is zero. For example for a node on the left side and one on the top side (5-12) and (5-13) are valid. For the outer mesh of the node on the top left side both apply [29].

$$V_{m,n} - V_{m-1,n} = 0 \tag{5-12}$$

$$V_{m,n} - V_{m,n+1} = 0 \tag{5-13}$$

We now can solve the system of linear equations built above and find the potential in nodes. Here, a technique is discussed that improves the conversion of the solution of this system of equations. In fact electron density (n) used in the above system as input can be written as:

$$n_{m,n} = N_C \Im_{1/2}[\frac{(F_n)_{m,n} + qV_{m,n}}{k_B T}] \tag{5-14}$$

F_n is quasi-Fermi energy, V is the old node potential, $\Im_{1/2}$ is Fermi-Dirac integral of order ½ and N_c is effective density of states in the conduction band. Looking at (5-9) we see that an increase in potential, implies a decrease in electron density, but in (5-14) it is the opposite. So using (5-14) along with Poisson equation in each iteration will limit the jumps of potential caused by solving Poisson. This will stabilize the convergence of Poisson equation and make the algorithm more efficient. (5-14) is nonlinear, so in a way we have a nonlinear Poisson system of equations now. We can use a Newton-Raphson loop to solve

it. The Jacobian matrix of this solution is a very sparse matrix which saves some memory and time [29].

Here we discuss application of NEGF to the transport equations. In ballistic transport, NEGF method is equal to solving Schrödinger equation with open boundary conditions.

First, we will see the algorithm to solve Schrödinger equation in 1D. This is done by an effective mass approximation of Schrödinger's equation. We want to review its solution in 1D, in the direction normal to channel length (across gates and channel) [29].

$$-\frac{\hbar^2}{2m_z^*}\frac{d^2}{dz^2}\Psi_i(z) - qV(z)\Psi_i(z) = E_i\Psi_i(z) \tag{5-15}$$

E_i is bound state energy of sub band i and $\Psi_i(z)$ is the envelope function of subband$i$. q is elementary charge and m_z^* is electron effective mass in z direction. From here bound state energies are calculated and used in (5-16) and (5-17) (that are related to the structure of lattice) to find the electron and hole densities [29].

$$n_i = n_{2Di}\ln(1 + e^{(\mu - E_i)/k_bT}) \tag{5-16}$$

$$p_i = p_{2Di}\ln(1 + e^{(E_i - \mu)/k_bT}) \tag{5-17}$$

n_{2D_i} and p_{2D_i} and k_B are constants and μ is body Fermi energy. Then (5-18) is used to find the current in unit of width [29].

$$I_{Di}/W = I_{Oi}\{\Im_{1/2}[(\mu - E_i)/k_BT] - \Im_{1/2}[(\mu - E_i - qV_{DS})/k_BT]\} \tag{5-18}$$

I_{Oi} is a constant and $\Im_{1/2}$ is Fermi-Dirac integral of order ½ [29].

Deep scaling of MOSFETs has caused tunneling through oxide layer a problem to be considered in design. In fact insulators have a finite band offset in relation to the channel and this offset can be passed by some carriers. To calculate the tunneling current, an integral is taken of the probability function of electrons in oxide layer. So with oxide layers as thin as 1 nm, it is necessary to simulate electron penetration in insulator. To do this, when solving Schrödinger equation, zero boundary conditions must be put at the interface between insulator and contact, not at the interface between semiconductor and insulator [29].

So when the problem is solving Schrödinger in 2D for the grid shown in Figure 5.1, we can first slice the 2D surface in z direction and solve the 1D Schrödinger as explained above for each slice. Electron penetration in insulator can be considered too. Then the Hamiltonian (total energy operator that helps calculate time evolution of quantum states) of the system is written in terms of two wave functions in the direction of width of the device and in the grid plane. The Hamiltonians size is related to the number of nodes: $(N_x \times N_z)^2$, but with mode space approximation, its size becomes related to nodes only in x direction: $N_x{}^2$. This, of course reduces the time of calculations. To be able to use mode space representation, we must be able to suppose that [20], [29]:

$$\frac{d}{dx}\Psi_i^*(x,z) = 0 \tag{5-19}$$

In other words, the changes of potential in z direction must be negligible. Then it is possible to rewrite 2D Schrödinger equation as a 1D (along the x direction) equation in terms of just one sub band. In other words, equation can be solved for each sub band separately. So the Hamiltonian can be crushed into independent smaller parts. The above

assumption fits SOI technology very well, because in a very thin channel, it is less likely that potential changes along the z direction. For bulk technology space mode representation is not applicable [29].

The next step is writing retarded Green function as [29]:

$$G(E) = [EI - H[E_i(x), E_{k_j}] - \Sigma]^{-1} = [E_l I - H[E_i(x)] - \Sigma]^{-1} \tag{5-20}$$

In which $E_l = E - E_{k_j}$ is longitudinal energy and Σ is self-energy matrix. Using Green's function, a 2D electron density matrix is calculated for each longitudinal energy level, as [29]:

$$n(E_l) = \frac{1}{\hbar a}\sqrt{\frac{m_y^* k_B T}{2\pi^3}}[\Im_{-1/2}(\mu_s - E_l)A_S + \Im_{-1/2}(\mu_D - E_l)A_D] \tag{5-21}$$

Total 2D electron density matrix can be derived from (21). Then, to produce the 3D electron density, in each node along x direction, distribution function $|\Psi_i(x,z)|^2$ is calculated with total 2D electron density matrix. The result can be given to Poisson equation for a potential profile calculation [29].

Current can be calculated using [29]:

$$I(E_l) = \frac{q}{\hbar a}\sqrt{\frac{m_y^* k_B T}{2\pi^3}}[\Im_{-1/2}(\mu_s - E_l) - \Im_{-1/2}(\mu_D - E_l)T_{SD}(E_l)] \tag{5-22}$$

Total current is found integration of (5-22) over all valleys (E_l) [29].

Scattering is another phenomenon to be modeled in a MOSFET simulation. In a general view, scattering happens because of defects in lattice, impurities, rough insulator surfaces and high densities of carriers. More specifically, sources of scattering can be categorized

as: surface roughness scattering and electron scattering with impurities, phonons and other electrons [29].

Usually mobility (μ) is used to represent the ease of electron displacement in an environment. In quantum analysis, this can be done through the concept of state lifetime, meaning that lifetime of a state of an electron is the time before a scattering event causes the electron to go to another state [29].

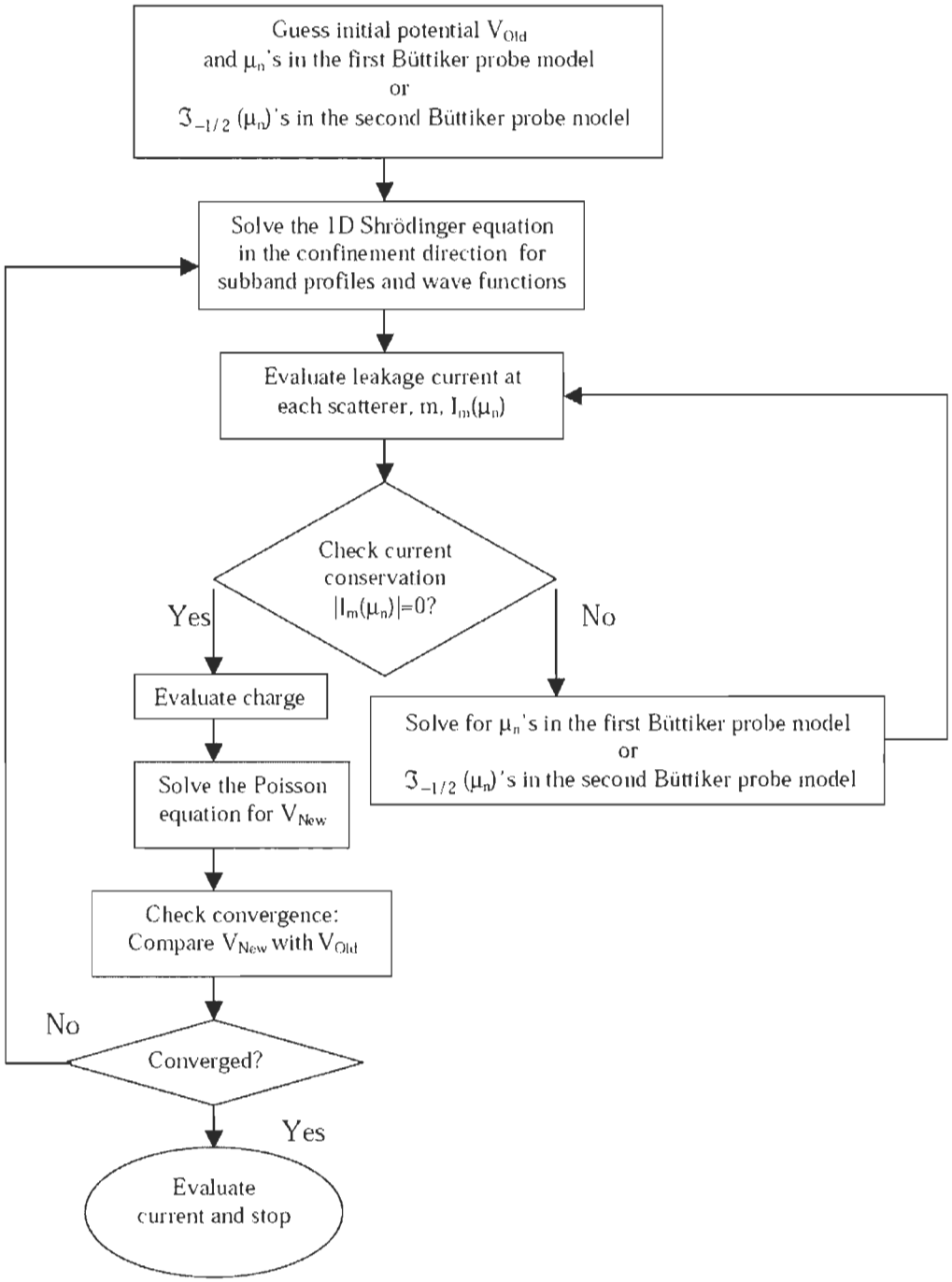
$$\mu = \frac{q < \tau >}{< m^* >} \quad (5\text{-}23)$$

q is the elementary charge and $q < \tau >$ is the average of electron state lifetime and $< m^* >$ is the average of electron effective mass in the transport direction. Averages are taken over different sub band energies that an electron might occupy [29].

The overall algorithm for simulating scattering is first to solve 2D Schrödinger equation as two 1D problems in vertical direction and in the channel direction. This will give the vertical electron concentration and sub band profiles and electron concentration in channel direction. Then having 2D electron density, 2D Poisson equation is solved to find new potential profile [29].

Physically, scattering means that the state of carriers is altered by some scattering phenomena while passing through the channel. The total number of carriers remains constant during these transformations. To model this, some functions are defined (called Buttiker probe) that take out some of the carriers from the system and import the same number of carriers, only with different energy states. This can mimic what scattering does to the carriers [29].

Figure 5-2, taken from [29] shows the process of self-consistent solution of Poisson and Schrödinger equation along with two Buttiker probes as methods of modeling scattering [29].



ZhibinRen, "NANOSCALE MOSFETS: PHYSICS, SIMULATION AND DESIGN", thesis, Purdue University, Octobre 2001.

Figure 5-2 Algorithm for modeling scattering

Using Boltzmann equation, source injects electrons into the semiconductor channel and they must pass over a potential barrier made by gate. In fact sub band energy of electrons must surpass a minimum energy that depends on gate potential. The densities of electrons in each subband on different sides of this barrier are described by two different equations (left is for source side and write is for drain side) [29]:

$$n_{left}(x) = \sum_i n_{2Di}\{\ln(1+e^{\mu_S}) + [\frac{1}{\sqrt{\pi}}\int_0^{E_{pi}} \frac{dE}{\sqrt{E}}\Im_{-1/2}(\mu_S - E) + \frac{1}{\sqrt{\pi}}\int_{E_{pi}}^{\infty} \frac{dE}{\sqrt{E}}\Im_{-1/2}(\mu_D - E)]\} \quad (5\text{-}24)$$

$$n_{right}(x) = \sum_i n_{2Di}\{\ln(1+e^{\mu_D}) + [\frac{1}{\sqrt{\pi}}\int_0^{E_{pi}} \frac{dE}{\sqrt{E}}\Im_{-1/2}(\mu_D - E) + \frac{1}{\sqrt{\pi}}\int_{E_{pi}}^{\infty} \frac{dE}{\sqrt{E}}\Im_{-1/2}(\mu_S - E)]\} \quad (5\text{-}25)$$

n_{2Di} is a constant, E_{pi} is the maximum energy of subband i, μ_S and μ_D are contact Fermi energies of source and drain and $\Im_{-1/2}$ is the Fermi integral of order -1/2 [29].

Using Green's functions, this density function is formed in one equation [29]:

$$n(x) = \sum_i n_{Oi}\{\int_{-\infty}^{+\infty} dE[\Im_{-1/2}(\mu_S - E)D_{Si}(E,x) + \Im_{-1/2}(\mu_D - E)D_{Di}(E,x)]\} \quad (5\text{-}26)$$

n_{Oi} is a constant and D_{Si} and D_{Di} represent the effects of source and drain on the local density function for subband i [29].

Ballistic current at source or drain contact is [29]:

$$I_D / W = \sum_i I_{Oi} \int_{-\infty}^{+\infty} T_{SDi}(E)[\Im_{-1/2}(\mu_S - E) - \Im_{-1/2}(\mu_D - E)]dE \qquad (5\text{-}27)$$

I_{Oi} is a constant and $T_{SDi}(E)$ is source-to drain transition in terms of electron energy. In classical analysis, $T_{SDi}(E)$ has a binary value. Above barrier energy it is equal to 1 and below that energy it is equal to 0, meaning that the barrier is considered as a solid wall. In quantum analysis, $T_{SDi}(E)$ has a continuous range of values above and below the barrier energy, meaning that there is always a possibility of passing or tunneling. Classical formula of (5-27) can be solved analytically, but quantum version is solved numerically and with iterations. [29].

Scattering centers are considered just like source and drain as sources of carrier. But they only change the state of carriers and not the number of them. The electron density function containing scattering effects is [29]:

$$n = \sum_i n_{Oi} \sum_j \int_{-\infty}^{+\infty} dE[\Im_{-1/2}(\mu_j - E) D_{ji}(E, x)] \qquad (5\text{-}28)$$

i represents subbands and j represent sources of scattering and carrier. D_{ji} is local density and μ_j is Fermi potential [29].

Current in each source of scattering or carrier I_{ej} is [29]:

$$I_{ej} = \sum_i I_{Oi} \sum_k \int_{-\infty}^{+\infty} T_{jki}(E)[\Im_{-1/2}(\mu_j - E) - \Im_{-1/2}(\mu_k - E)]dE \qquad (5\text{-}29)$$

i represents subbands and k represents sources of current. T_{jki} is transmission from k to j in subband i [29].

5.2 Algorithm establishment

NanoMOS has been created with a principally quantum physical and mathematical concern, rather than a coding approach. That is because the challenge in this quite new aspect of electronic device design is still finding solutions for physical modeling and solving complex models, Coding and optimization come afterwards. MATLAB has been chosen as the programming language for this tool in order to be able to benefit from advanced functions and routines that exist in this language [36].

NanoMOS has almost 4500 line of code distributed in different functions that are called according to the type of simulation chosen by user. Here These functions and routines are introduced briefly and then we will have a more detailed look at their position and functionality in the whole algorithm [36].

[36] divides the routines of NanoMOS to two group of big and small routines. This division is made in regard to the algorithmic importance of the routines. Big routines according to [36] are [28]:

poisson: it solves Poisson equation.

charge... routines: they calculate charge density for different transport models.

current... routines: they calculate current densities for different transport models.

et: it handles energy transport model.

current_mat: it calculates the Fermi level for scattering transport model.

schred: it solves Schrödinger equation in transverse direction.

saveoutput: it saves the data and plots them.

Main: it manages the flow of algorithm and calls other routines.

According to [36], small routines are [28]:

nmos_in: this is the function to call in MATLAB environment. It sets some input data and calls main and some other routines.

doping: it generates 2D doping profiles.

fermi: It computes Fermi integrals.

integral: it calculates classical charge density for classical quantum model.

fprime: it calculates Fermi-Dirac integrals for Poisson routine.

dummy and anti_dummy: they convert charge to quasi-Fermi level and vice versa for non-linear Poisson solver.

dummy_prime: it differentiates dummy functions.

Now let's follow the flow of algorithm as NanoMOS starts to work and performs a simulation. In this way all of the functions will be introduced in more details and the way they work and relate to each other will be discussed. In this process we mainly follow the calculation algorithm and neglect I/O coding.

As it was mentioned, this tool gets started by typing "nmos_in" in MATLAB. nmos_in.m is the run file and input deck of NanoMOS. All the input variables are set here and then the necessary functions are called. After defining some global variables, nmos_in calls phys_constants.m which simply sets some basic physical variables like ε, permittivity constant and q the elementary charge [28].

After that, nmos_in includes input variables discussed in last sub-chapter of chapter 4.The first set of input is transport variables, including transport model, semiconductor material, and mobility model. Silicon with different lattice orientation, Germanium and some III-V materials are among material choices. There are 11 choices of material in total. Then bias and device geometry variables, as discussed in sub-chapter 4-3, are set [28].

Having this input data,nmos_in calls emass_cal and passes the material chosen by the user as input to this function. emas_cal assigns uses a simple MATLAB switch function to choose between different sets of material physical data according to its input, material [28].

Then nmos_in continues to set mobility, grid, quantum parameters, solution and optional input variables. Grid spacing in x and y directions is set to 0.3 and 0.1 nm as default. It is interesting to pay attention to the fact that grid is more finely defined in y direction. This helps model more significant potential gradient in y direction. It is also noticeable that criterion_outer and criterion_inner (two parameters defining the convergence limits of inner and outer loops) are set at 1e-3 and 1e-6 [28].

Now nmos_in has defined all the input and can call main.m. main.m does all the computation and returns the desired parameters. Nmos_in times the execution of main.m to show the computation time of the simulation at the end. When main returned its output variables, nmos_in will save the output in output.mat and then call saveoutput.m. saveoutput.m plots the data as desired by user and does the post processing part [28].

Now we look at main.m and follow its different branches. main.m sets a variable named "progress" to keep track of progress of the algorithm and show it to user as it finishes each part. Then it starts "initializing". During this process it first makes sure that criterion_outer is not too little for some transport models. Then it calculates total number of bias points and

Then it creates two vectors containing gate and drain voltages for all the bias points specified by user. Then it rearranges grid so that there is an integer number of meshes in each region (source, channel, drain) of device. Then it assigns some parameters that will be used in quantum equations (doping densities...). Finally it calls device_geo.m to draw the plan of the device with grid on it. This way main.m finishes its initial preparations and enters to the process of calculating some initial parameters [28].

First it allocates memory to some matrix parameters that will hold the key physical values of the device. These parameters will be used in the following parts and will pass some data to other functions. Some of them, like electron density and potential profile, are 1D vectors and have a value for every grid point. Others like subband energies and electron density for each subband are 3D matrices with values for every valley in every subband in every grid point along x direction. Current is predicted to have one value for every bias point [28].

Then it calls doping.m. It was mentioned in sub-chapter 4-3 that Gaussian doping profiles are supposed for source and drain whose slopes are determined by user. doping.m generates the functions of doping profile according to the input chosen by user over the grid. In fact, doping.m assigns a doping value to every grid point. Then main.m calls fprime.m which calculates the derivative of doping and saves it in a matrix. This matrix will be used by Poisson solver [28].

From now on, main.m starts self-consistent solution of Poisson and Schrödinger functions. First it calculates mobility in relation to doping profile. To do so, it calls mobility.m and passes mobility model number to it. According to the temperature, electric field, material and mobility model, mobility.m calculates the mobility of channel.

NanoMOS uses MATLAB "sparse" function to compress big matrices and increase the speed of calculations. At the end of computations it retrieves the full matrices using "full" function and then sends them to nmos_in.m [28].

Then main.m makes its initial guess. To do this, chargenanomos.m is called. chargenanomos.m is the routine that calls charge and current routines that solve transport models. For the initial guess, chargenanomos is called with transport model 1 and a fixed initial value for drain voltage. Input variables that are passed to chargenanomos by main.m are old electron density, old potential energy profile, old subband energies, old electron densities or subbands and if it should calculate charge or current. Chargenanomos, on the other hand, returns new electron density, subband energies, new electron densities for subbands, current density, current in each subband and electron temperature for subbands [28].

When chargenanomos is called for initial guess, it calls charge_init.m, which sets initial values of electron density and potential profile for grid points. No transport equation is used to this point [28].

When main.m receives initial guess for charge density and potential profile, it calls poisson.m. poisson.m gets doping density, electron density and old potential profile vectors and produce new potential profile for grid points in a vector. poisson.m solves Poisson equation in 2D using Newton method. To do so it uses fermi.m function that takes Fermi integrals of desired order. poisson.m has a loop that satisfies a convergence parameter, criterion_inner, that is compared with the change in potential profile in each iteration[28].

Then main.m enters what is called "current calculation loop". But actually it solves transport model and Poisson equation self consistently and calculates all the desired energy profiles and charge densities in each bias point. It also calculates I-V characteristics [28].

Here, main.m enters a "for loop" on gate voltage. Number of increments depends on number of gate voltage steps chosen by user in input document. Here only a vector is created that contains gate bias voltages and immediately the second loop starts for drain voltage. Its increment number depends on number of drain steps chosen by user. Here too a vector is created that contains drain bias point voltages. Inside these two loops or in other words for each bias point, there is a while loop to continue a self-consistent solution of Poisson and Schrödinger until a convergence factor, criterion_outer is satisfied [28].

Inside this while loop, three main actions are taken: first chargenanomos.m is called to calculate charge density parameters according the transport model. As explained above, chargenanomos does it by calling one of charge...m functions. One of these functions exists for each transport model. Bellow we will review their main functionalities and differences. When chargenanomos returns new charge densities, main.m calls poisson.m which computes new potential profile. Then the maximum difference between old and new potential profile vectors (happening at any grid point) is compared with criterion_outer. If it is bigger than criterion_outer, the loop continues until the responses of two equations converge to each other and the change in potential profilein one iterationbecomes very small. When this is achieved, while loop is ended and convergence data is saved [28].

With the final potential profile at hand, main.m calls chargenanomos one last time to get the final charge density parameters. Then chargenanomos.m is called again to calculate current. To do this, it calls one of current...m functions according to transport model.

Bellow these functions are explained. Having calculated current for the present bias point, all of the calculated data, related to the present bias point is saved in separate matrices. Then the inner for loop is incremented and the whole process of self-consistent solution repeats for the next bias loop. When all of bias points have been treated, main.m finishes. The variables that main returns are: current density, current density in subbands, electron temperature in subbands, electron density, electron density in subbands, conduction band profile, subband energies, Fermi energy level and convergence data [28].

Now we will have a quick look to charge and current computing functions. There is a charge density computing function for each transport model. They need to solve the charge density problem in two dimensions. All of them solve the Schrödinger equation in confinement direction (gate to gate direction) and then apply their respective transport model in the channel direction and produce 2D results. The function that all of them call to solve Schrödinger equation in 1D is schred.m. schred.m takes potential energy profile as input and produces subband energies and wavefunctions in confinement direction. Current calculating functions act similar to charge computing functions (after they have received the results of schred.m). They just apply the formulas related to their respective transport model to the potential profile and other input data and calculate current and carrier temperature [28], [36].

Knowing the exact functionality of each routine and their role in total algorithm is necessary to consider the possibilities for parallelization of the algorithm. In the following section some of these possibilities will be presented and the parallelization algorithm implemented in this work will be discussed in more details.

5.3 Parallelization

To parallelize an algorithm, one should look for parts of the algorithm that can be executed separately at the same time. These portions of the algorithm must be independent of each other, meaning that their development must happen without relying on each other's results. It is possible to establish connection between different processors, using Message Passing Interface (MPI) or other means, but it is a sensible operation and different parameters must be considered. Of course one needs to make sure that developed parallel algorithm will work correctly in all situations, meaning that different processing unites, communicating with each other, will send their respective messages at about the right time, otherwise improvement factor suffers. Also in HPC machines data transfer and communication is a much more time consuming process than data processing. So by dramatically increasing communication in an algorithm, advantages of parallelizing may be lost. Choosing the best parallel algorithm in this sense depends partially on choosing routines to be parallelized that need the least communication and also on distributing the job between as many parallel processors a possible, without letting data transfer time overcome the advantage of greater distribution.

Here, I will try to identify and discuss different possible strategies of parallelization of NanoMOS code. At the end the chosen strategy is explained in more details.

NanoMOS, in brief, is a code that performs a series of computations and equation solving algorithms on a grid of space nodes for each bias point (and in some routines each subband and each valley). From this structure some parallelization strategies emerge instantly. First of all it seems promising to try to distribute the computation over grid nodes or bias points. Then, the computation algorithm for each grid node in each bias point can be

examined. Possibly it offers some parallelization opportunities too. In all of these considerations, independence of computations in parallel tasks is the first question to be asked [28].

Parallelization over bias points is a very straightforward and at the same time heavy solution. It is straightforward, because it has a lot of algorithmic clarity. In NanoMOS the main computation loop (self-consistent solution of Poisson and transport equations) is repeated for each bias point. It is heavy, because in this strategy, all of the computing routines are engaged in parallelization. Each one of them must be examined for compatibility with parallel algorithm, their variables must be passed and returned in order and correctly and finally more errors must be dealt with when a large number of routines are involved in some code modifications. The computations related to bias points are independent from each other. They only need some common initializations that can be done outside parallel part of the code [28].

For parallelization over grid nodes some initial decisions need to be made. In this approach, modifications must be made in each routine that works on grid nodes. Each routine takes its input as large matrices, containing data for all of the nodes, performs its computation on the matrices and returns its variables as matrices too. So parallelization has to be done inside the routines, separately from parallelization in other routines. This increases the overhead computation to some extent. There are many routines working with bias points. Computations of each node in these routine is generally independent of computations of other nodes. To choose some of these routines as candidates for parallelization, they should be checked to see if a large part of their computation is done inside loops with grid nodes as index or not. For example fprime.m and schred.m functions

have long for loops (for loops with a lot of computation lines) over grid nodes, poisson.m and charge...m and current...m functions have short for loops (for loops with a few computation lines) and doping.m falls somewhere in between. Now if the coding language or parallel function requires a setup of parallel task with a large overhead each time a loop is being distributed, parallelizing short for loops may not be beneficial. But fprime.m is not part of self-consisting lop, so parallelizing it will not decrease the computation of the whole code very much. It should also be noted that the amount of computation for each grid node in any of the above applications is very little and not enough to be sent to one computational unite. Also the number of nodes is much bigger than most cluster sizes. That is why it seems necessary to divide the whole grid nodes into a number of groups and send all the grid nodes in one group to one computing unit in cluster [28].

As for parallelization over valleys and subbands, charge...m and current...m functions must be considered that have quite long for loops over these two parameters. The important point to consider here is that these functions are transport equation solvers. They are part of self-consistent solution and are coupled with Poisson solver. If only subbands and valleys are taken as parallelization parameters, Poisson equation and parts of transport equation solution (schred.m) are still being computed completely serially. The most compelling solution seems to be parallelizing over grid nodes in functions in which it is beneficial and parallelizing over subbands and/or valleys in transport model solvers [28].

By looking at computation algorithm, of course, any solution must include self-consistent loop. But the two main computational blocks inside the loop (Poisson and Schrödinger solvers) are dependent on each other's final results, so they cannot be parallelized. If we look inside these two blocks, we would see that poisson.m does not offer

any algorithmic possibilities, but chargenanomos.m, depending on transport model, may offer some. For example, inside charge_dd.m there are some parallelizable parts of algorithm. Inside the "for loop" for valleys, the loop over subbands and the rest of computations seem to be independent and can be parallelized. This, by itself, is not a significant improvement, However it can be part of an extensive parallelization strategy [28].

Now let's make some conclusions about different strategies. When there are more than one bias point, parallelization over bias points is very efficient. Because in this strategy, parallel section includes almost all of computational load and there is not heavy communication overhead. Each parallel unit communicates with central node only at start and the end of its computation. Communication includes quite large matrices, but it is the same for other strategies too. This strategy needs dealing with a lot of functions and variables. It is more difficult to make sure that the whole algorithm of self-consistent solution can be run for all of the bias points simultaneously. Different parallel threads must be able to call the same functions at the same time and use common sets of data and variables. When the goal is to parallel the computation for just one bias point, different solutions should be employed at the same time, as explained above. To include as much computation load as possible in the parallel algorithm, each function has to be parallelized over the variable that works best for it. Doing this in each function is simpler than parallelization for the whole algorithm, as in the case for bias points, but on the other hand it involves various parallel operations [28].

Parallelization of an older version of NanoMOS code is explained in [37]. This work has been done using MPI and PVM parallel toolboxes created for MATLAB by

another researcher ([6]). This work differs in two major aspects from my project. First, it was performed on a detailed physical model that took a much longer time to be simulated. Second, it was implemented with different software and hardware tools. At the time that this work was done, MATLAB was not directly parallelizable and they used some additional toolboxes, written for MATLAB Also the HPC facility used in this work is different from one used in current work [37].

In this project, NanoMOS2.5 is parallelized over bias points. A series of bias points with equal gate voltages and different drain voltages are used for simulations. "parfor" function of MATLAB is used to parallelize self-consistent loop of NanoMOS. Then the parallel code is run on a cluster of processors. The results are presented in the next chapter [28], [39].

As it will be explained in more details in the next section, "parfor" is used to parallelize the self-consistent solution of potential and charge density solvers for different bias points [28], [39].

Parallel coding in MATLAB is done using MATLAB Parallel Computing Toolbox, which includes some parallel functions and is being increasingly integrated with some other toolboxes of MATLAB to be able to support parallel computing in data processing in different toolboxes. MATLAB Distributed Computing Server (MDCS) is used to connect a cluster of computing nodes that are managed by a scheduler or a job manager. MDCS comes with MATLAB job manager that can be installed to manage the cluster, but it can be linked to and work with third party schedulers too. Each computing node in cluster (worker) has a license of MDCS and a MATLAB session runs on it with any toolboxes

necessary for the computation. The main node (client) has the Parallel Computing Toolbox installed on it and communicates with the cluster through job manager [39].

On the Krylov system, there is a procedure for running MATLAB parallel codes. The user must save the main MATLAB code containing the parallel functions and all of the MATLAB files that are called by this main file on the hard disc on the remote computer (Krylov system). Krylov has its own license of MATLAB Parallel Computing Toolbox (MPCT) and MATLAB Distributed Computing Server. The user only provides the code files generated using their own licenses. Krylov also has its own scheduler which is connected to the MDCS. MDCS receives parallel jobs from MPCT and communicates accordingly with the scheduler of the system to create the parallel platform needed for that certain parallel job to be distributed on. In this way, on Krylov, the user does not need to deal with creating the parallel platform themselves. For example when a parfor is used, the user only determines the number of workers to be created by the MDCS by choosing the loop parameter of parfor. MDCS and scheduler will take that number and create an array of computers with the same number of workers to run the parfor loop and a client computer to run the main code. The communications between the workers and the client is also managed by MDCS and scheduler. In a parfor application, these communications are already implemented in the definition of parfor and automatically performed when the parfor is being executed [22], [39].

5.4 Implementation

In this section, the steps that have been taken in implementation of the algorithm described in previous section are discussed. This section has a very practical nature. It has little theoretical content and is not meant to be taken as a reference on coding or algorithm

development, but instead it might provide a more guided experience for students following a similar path and trying to make parallel modifications in MATLAB environment.

When a "parfor" loop is used in a MATLAB code, some basic rules are checked by MATLAB before running the code. If a MATLAB code, containing "parfor", is run without opening the "matlabpool", MATLAB runs the code in serial and executes "parfor" loop as a for loop. Nevertheless, it checks some rules that are specified to "parfor". These rules are laid down to make sure that "parfor" is understandable by MATLAB and that the loop body is distributable to different workers. One of these rules is about classification of variables. MATLAB accepts four types of variables inside a "parfor" loop [39].

The first problem, as it apparently happens in many cases, appeared with variable classification. A matrix can be indexed inside a "parfor" loop by a loop counter, but any other first level index of that matrix has to be a constant, a non-loop counter variable or a colon. The goal of this rule is that MATLAB needs to be able to distribute only necessary part of the variables to each worker at the time of distributing parallel tasks (sliced variable) [39].

Any matrix indexing that violates this rule, create an error. In the case of incorrectly indexed matrices, the indexing has to be broken. First it has to be used only with one of the indexes, and then the other one. There are examples in documentation that explain this. In NanoMOS code this happened with variable "conv" at the end of main.m. This is not a clear case of "nested for loop", but apparently "`1:length(converge)`" causes the same indexing problem [28], [39].

```
conv(1:length(converge),ii_vg,ii_vd) = converge;
```

[28]

To solve this, "converge" is saved as in an indexed matrix inside the "parfor" loop:

```
myconverge(:,ii_vd) = converge;
```

[28]

And then, outside the "parfor" loop, it is put into "conv":

```
for (cnv_count = 1:Nd_step+1)
conv(:,ii_vg,ii_vd) = myconverge(:,ii_vd);
end
```

[28]

Another type of error was references to variables that were considered as cleared variables. This happened when a variable was defined and assigned a value before the "parfor" loop and inside main.m or one of other functions. Then when the variable was used inside the "parfor" loop, it created an error [28].

No clear explanation of the reason for this error was found in documentation, but I believe it can be explained with the concept of workspaces in MATLAB. "Workspace" is a set of variables related to a session of using MATLAB that are kept in memory while that session continues. But different workspaces might exist at the same time. Variables used in each MATLAB function form a separate workspace, so normally these variables are not accessible from other workspaces [39].

main.m is a function called by nmos_in.m. main.m, in turn, calls some functions and passes some variables to them. When "parfor" loop is started, different workers are created by MATLAB which work with variables that might have been assigned some values

before, inside main.m. The problem is that the definition of these variables or their value assignment might be before the start of "parfor" loop. In this case, workers might find these variables not correctly defined.

To solve these sorts of errors, it might be necessary to redefine some variables, or reassign desired values to some variables inside the "parfor" loop too. In doing this, we must pay enough attention not to ruin algorithm of calculations.

Another important type of error that might happen in the process of modifying the code is that functions that are called by main.m do not return their variables correctly. MATLAB documentations about "parfor" do not discuss calling functions from inside a "parfor" very thoroughly. There is actually little notes on calling nested functions, but no clear cautions about calling functions. This, normally, would mean that there should be no problems with calling functions. But actually some errors are created that say the returned variables are not assigned. This might be due to the fact that different workers are calling the same functions and working with the same variables at the same time. Naturally, function executions and function output variables should be separated on each worker, but when there are function calls inside the loop, there seems to be some sort of separation problem. It seems to the writer that such complications are not very hard to imagine when a very high level language is adapting parallel computing. In any case, a solution to this problem can be replacing a function call by actual code of that function inside the loop [39].

This solution has been employed to solve this problem in a very simple and effective way. Chargenanomos.m and poisson.m and function calls inside these functions have been replaced. This solution necessitates some considerations that will be discussed below.

When code lines have replaced functions, function workspaces mix. It means that variables that were being created and used in different workspaces are now being seen by the worker in one workspace. This makes a second round of variable checking necessary. This time it should be checked if variable name similarities exist between functions and if supposedly separate uses of variables in functions are being invaded after code replacements. These sorts of errors and algorithm malfunctions, when found, can be easily corrected by renaming variables.

Sometimes there are parts of code that are not compatible with a parallel implementation. These parts of algorithm have to be rewritten or eliminated. In main.m there is a variable called "progress" that keeps the percentage of completion of the whole simulation. "progress" is incremented after each bias loop is finished and then it is printed for the user to see the progress of simulation. When a loop is distributed between workers, the linear concept of loop progress does not exist anymore. The progress is happening inside the workers and the client has no idea of the progress of each worker until it receives the results from workers. At the beginning of the loops over bias points, there are a few lines that check if all bias points have been treated and increments "progress" and prints it. Since this part is not compatible with parallelization and it is not a crucial part og the algorithm (it has no computational value), it was eliminated from the parallel code [28].

The last point to be mentioned is the "route of execution". NanoMOS calls different routines according to the user's choices of transport model, electron penetration limit, etc. More specifically, different Schrödinger solvers are called according to transport model and each one of them call a different set of functions. Also in most of the routines, different routes are taken according to the state of some flags set by user. These flags are electron

penetration, mobility model, etc. Because the measurements on the improvement factor of parallel code was to be done using a fixed simulation setup (a fixed set of input data), all of the code replacements were done according to the route corresponding to a certain input set. To do simulations with different setups, the corresponding route must be carefully identified and necessary code replacements should be done along that route [28].

To give a practical reference of the function calls in NanoMOS code, the "branching plan" of NanoMOS code is given here. The route corresponding to each transport model can be easily found using this list [28].

nmos_in: calls main, phys_constants, emass_call and saveoutput.

Main: calls device_geo, fprime, doping, mobility, chargenanomos and poisson.

chargenanomos: calls anti_dummy and the following functions for the mentioned purpose or according to mentioned transport model. The functions called by each one of charge... and current... functions are mentioned in parantheses. Chargenanomos:

For initialization, calls charge_init (does not call any functions).

For "drift diffusion transport" model, calls charge_dd (schred, mobility, anti_dummy) and current_dd (mobility, anti_dummy).

For "semi-ballistic transport" model, calls charge_semib (schred, fermi) and surrent_semib (fermi).

For "quantum ballistic transport" model, calls charge_qbte (schred, myquad, fermi, func_energy) and current_qbte (fermi, func_energy).

For "energy transport" model, calls charge_et (schred, et) and current_et (et).

Poisson: calls dummy and dummy_prime.

emass_call: calls emass which is written inside the same m file.

func_energu, dummy and dummy_prime call fermi.

phys_constants, fprime, device_geo, doping, mobility, anti_dummy, schred, et, fermi, myquad do not call any function.

[28]

Chapitre 6 - Validation

6.1 Comparison of serial and parallel code performances

General goals of any parallelization are making the computation feasible (mostly when very large data sets are involved) and making the computation faster (both when very large data sets are involved and when the computation is very time consuming). In this project, shorter computation time is the main goal. But speed-up factor has its own limits. Amdahl's law (6.1) explains how non-parallelizable portion of an algorithm limits the speed-up factor [20], [40].

$$S = \frac{1}{1 - P} \tag{6.1}$$

S is the speed-up factor in relation to serial algorithm and P is the parallelizable percentage of algorithm. Amdahl's law ignores communication lost time and gives a maximum limit of S for a large number of processors. Gustafson's law includes a finite number of processors in calculation of maximum S [41]:

$$S(P) = P - \alpha(P - 1) \tag{6.2}$$

S is speed-up and P is the number of processors and α is the part of the algorithm that is not parallelizable [41].

As it was explained in previous chapters, in a simulation with multiple bias points, in a one to one comparison, parallelization of NanoMOS over bias points is the most

inclusive strategy, in the sense that it leaves out the shortest "critical path" (the longest chain of dependent calculation) out of parallelization process and does not add anything to it, While other strategies include serial execution of a larger part of algorithm [20], [28].

Parallel code was run on 2, 4, 8, 16 and 28 workers on Krylov cluster of CLUMEQ. Currently, Krylov's limit for the number of workers is 32. But it is not possible to run the code using all of the accessible units as workers, because at least one of the processing units has to take the role of client and there might be other limitations too. Bias points are distributed with drain voltage following a simple step function over the range of 0 to 1.62 V with 0.1 V steps. In the case that drain voltage exceeds 1.62 V, The step size is reduced (to 0.06 V for 28 workers). It is worth mentioning that computation time in each run has a tolerance. Usually in sequential runs the tolerance is small, but over a longer period of time this tolerance might grow, which makes it difficult to have an exact reading of the computation time for different simulations. To minimize the effect of tolerance, each measurement was repeated a number of times. Then unacceptable results were eliminated and an average of acceptable results was used as the computation time for each number of workers [22], [39].

In Figure 6-1, speed up results (execution times of serial code normalized for parallel code execution time) are presented for different numbers of workers:

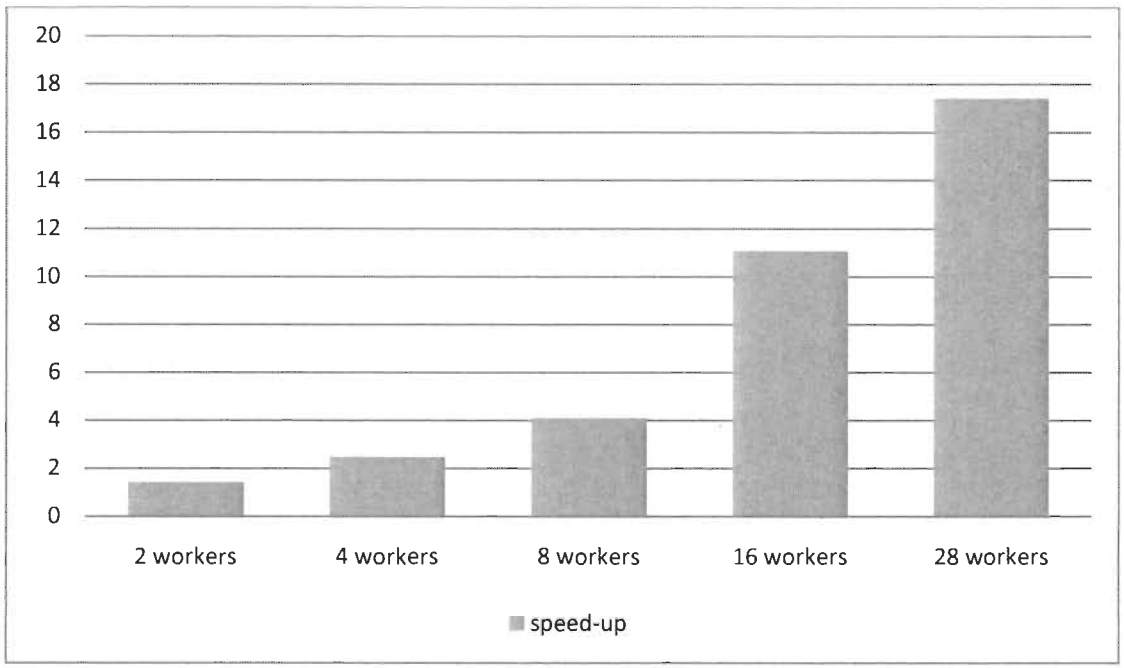


Figure 6-1 speed-up for different numbers of workers

It is well expected to see a dramatic decrease in execution time as number of processors increase. By increasing the number of workers speed up will increase too. This algorithm, as discussed before, does not saturate very soon as the numbers grow, because there is not extensive communication involved.

6.2 Validation

Quantitative validation of this work is fairly simple, because the parallel algorithm does not affect quantitative aspects of the algorithm. In other words, computations are performed in the same way in series and in parallel case. To verify this claim, convergence data (discussed in previous chapters) of simulations were compared. In Table 6-1 logs of convergence data for two parallel simulations with two and four workers (consequently with two and four bias points) are presented.

Table6-1 Convergence data for simulations with two and four workers

Simulation1		Simulation2			
Bias point 1 (0.0 V)	Bias point 2 (0.1 V)	Bias point 1 (0.0 V)	Bias point 2 (0.1 V)	Bias point 3 (0.2 V)	Bias point 4 (0.3 V)
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
0.0222	0.0222	0.0222	0.0222	0.0329	0.1107
0.0421	0.0392	0.0421	0.0392	0.0381	0.0377
0.0375	0.0339	0.0375	0.0339	0.0327	0.0320
0.0261	0.0225	0.0261	0.0225	0.0214	0.0207
0.0153	0.0134	0.0153	0.0134	0.0130	0.0128
0.0101	0.0098	0.0101	0.0098	0.0096	0.0095
0.0076	0.0075	0.0076	0.0075	0.0074	0.0074
0.0059	0.0059	0.0059	0.0059	0.0058	0.0060
0.0046	0.0045	0.0046	0.0045	0.0045	0.0046
0.0035	0.0035	0.0035	0.0035	0.0035	0.0035

0.0027	0.0026	0.0027	0.0026	0.0026	0.0026
0.0020	0.0020	0.0020	0.0020	0.0020	0.0019
0.0015	0.0015	0.0015	0.0015	0.0015	0.0014
0.0011	0.0011	0.0011	0.0011	0.0011	0.0011
0.0008	0.0008	0.0008	0.0008	0.0008	0.0008

Two columns of the first simulation are in order equal to the first two columns of the second simulation. This clearly shows that two simulations follow the same computational sequence when the input data (here bias points) are equal.

6.3 Parametric analysis

Communication is an important factor in any parallelization strategy. As explained in the discussion about grid distribution strategy, it is not always beneficial to distribute a job among as many processors as possible. With a large number of processors, and little computation for each one to do, communication between the main node and computing nodes (to send data and to initialize the node) plus communication between nodes (as part of computing algorithm) might even out the time saved by distributing the computation. Depending on the algorithm and amount of communication there might be a maximum number of computing nodes in which speed-up is maximum and after that, by increasing the number of computing nodes, communication overcome the effect of parallelization. If limitations of resources and economic efficiency are considered too, the maximum number

of computing nodes might be even smaller that the number determined by the algorithm. In this work, because there is no communication between workers and there is little communication between client and workers, there seems to be no limit for speed-up.

To see the effect of the number of workers on efficiency of parallel code, the speed-up for different numbers of workers is presented along with Gustafson's law's speed-up prediction in Figure 6-2. Taking Gustafson's law's speed-up as the theoretical maximum value of S that is decreased to a practical value because of communication lost time and other delays, it is expected that Figure 6-2 gives us a measure of the impact of number of workers on efficiency of each experiment. Non-parallelizable portion of the algorithm has been estimated by measuring the computational time of "initializing" part of NanoMOS in relation to the total computational time [28], [41].

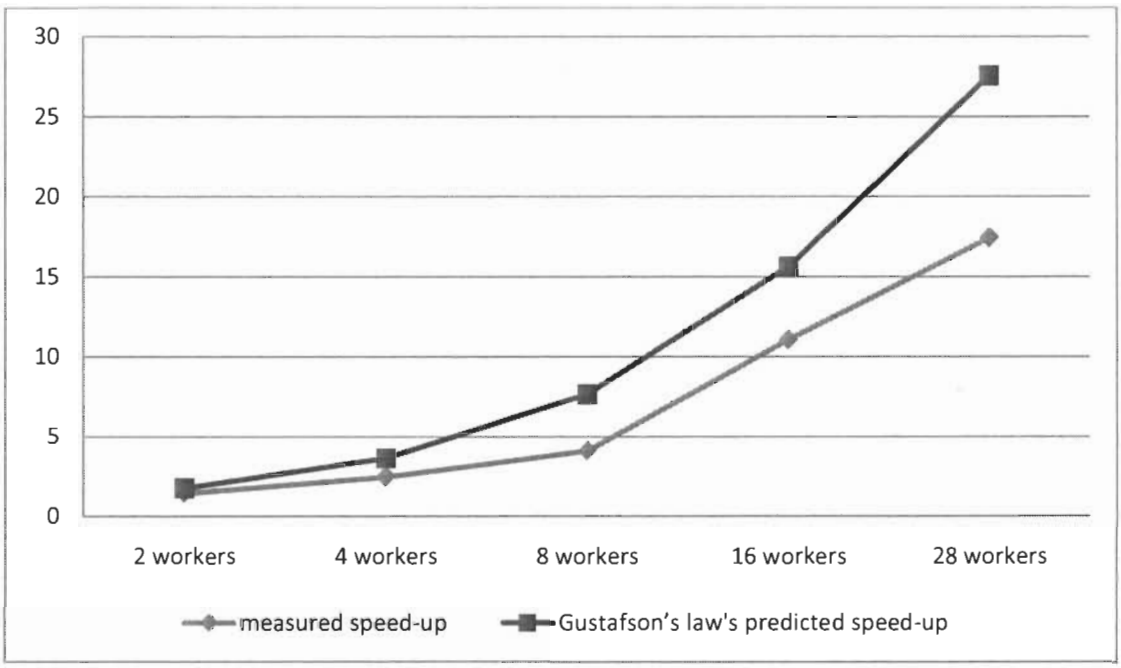


Figure 6-2 Measured and theoretical speed-up

To see the relationship between two lines more easily, measured speed-up is normalized to Gustafson's law's prediction in Figure 6-3.

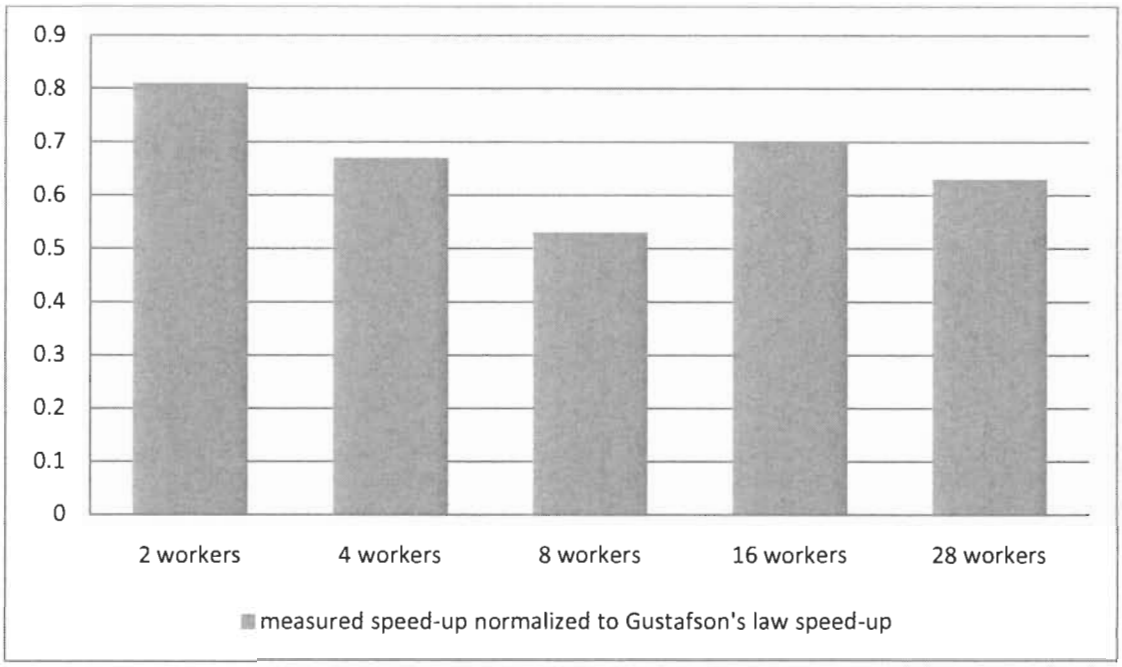


Figure 6-3 Measured speed-up is normalized to Gustafson's law's prediction

It is difficult to observe a very obvious pattern in Figure 6-3. This is mostly due to the fact that parallel code run time is relatively small and so tolerances have a considerable effect on it. As a result calculated speed-up is strongly affected. Nevertheless a slight downward slope is visible in Figure 6-3. In other words, as number of processors increases, speed-up factor follows the Gustafson's law less closely. This is because the difference in computational time for different bias points. Higher drain voltages impose longer computational time. In parallel execution of the code, the workers that finish their computation sooner, will return their results to client and then they will wait until workers that are computing bias points with higher drain voltage (hence with longer computation time) finish their job. It is clear that this means non-ideal use of resources. In other words, parallelization is not ideal as it is supposed in Gustafson's low. Computation times for drain voltages near to zero are quite close. Then when drain voltage becomes bigger (when there

are more bias points and more workers) computation time grows and the difference between workers run time increases. That is why measured speed-up is closest to theoretical prediction in the experiment with two workers.

Chapitre 7 - Conclusion

MOSFET scaling has continuously made higher processing power with lower price in smaller space possible for decades. Such a high yearly improvement factor, kept for such a long time, is by far an industrial exception. This dramatic and constant improvement has been realized to a great deal due to MOSFET scaling. And it is important to note that it has not happened without tireless efforts to go beyond technological limits of each time. In this sense, today's efforts in MOSFET scaling are a part of a long lasting trend. But in relation to MOSFET as a specific device, it seems that we are at a turning point.

As MOSFETs are scaled down to nanoscale, new physical effects become important in their performance. Due to very small dimensions in today's devices, these effects have quantum nature. In some specific aspects of the device design, for example doping concentration and oxide layer thickness, we are already counting atoms; in other aspects and parts it may happen not in a very long time.

In the long term, transistors might go under dramatic changes. Quantum transistors and molecular transistors are examples of possible candidates for post CMOS era. While there is still a long time until these technologies become available and MOSFET technology has still a considerable amount of improvement to make, it has been suggested that MOSFET scaling is starting to face some fundamental physical barriers and MOSFET technology is reaching its scaling limit.

Whether or not we consider future transistors, quantum transistors for instance, a continuation of MOSFET concept, it seems crucial to extend current microelectronic science to quantum and molecular scales. In fact, not only future MOSFETs will work under these new rules, but also future transistors will do so too.

It is necessary to model these quantum phenomena effectively, if we want to be able to design MOSFETs for beyond 10 nm regimes. Exact simulation of these effects can be very time taking. Finding ways to carry out these computationally expensive simulations seems important for future designs. High performance computation (HPC) facilities create an opportunity for such simulations to be done in reasonable time and with desired precision.

Another important scaling tool is novelty in geometry. New geometries decrease unwanted effects and open way for more scaling while keeping performance at an acceptable level. Double gate MOSFET has been suggested as an effective solution to short channel effects.

NanoMOS, an open source tool developed by researchers at Purdue University, is a simulator of double gate MOSFETs in nanoscale. It considers ballistic transport in MOSFETs and solves Schrödinger and Poisson equations to find charge distribution and current.

In this work, NanoMOS code has been studied for parallelization possibilities and a parallelization algorithm has been implemented on it. The result has been tested using HPC facilities of Compute Canada. Simulation results and time improvement factors have been presented and discussed for different numbers of processing nodes. This work, as a parallelization experience, can be used in future to implement parallel algorithms on more

complex models, making it possible to perform parametric studies and model optimization and achieve a precise design strategy for nanoelectronic devices.

References

[1] Shakouri. A., "Nanoscale Thermal Transport and Microrefrigerators on a Chip", *Proceedings of the IEEE*, vol. 94, Issue 8, pp 1613-1638, Aug. 2006.

[2] Yong Zhan, Goplen B., Sapatnekar S.S., "Electrothermal analysis and optimization techniques for nanoscale integrated circuits", *South Asia Pacific Conference on Design and Automation*, 24-27 Jan. 2006.

[3] Ble, S.V., Skorek, A.W., "Parallel Approach to the Nanothermal Numerical Analysis", *Canadian Conference on Electrical and Computer Engineering, CCECE '06*, May 2006.

[4] Liu, W., Asheghi, M., "Impact of phonon-boundary scattering and multilevel copper-dielectric interconnect system on self-heating of SOI transistors",*Twenty First Annual IEEE Semiconductor Thermal Measurement and Management Symposium,* 15-17 March 2005.

[5] Pop, E., Goodson, K.E., "Thermal phenomena in nanoscale transistors",*The Ninth Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems, ITHERM '04*, 1-4 June 2004.

[6] Issa, M., Skorek, A.W., "Nanoscale Thermal Analysis of Electronic Devices", *Canadian Conference on Electrical and Computer Engineering, CCECE '06*, May 2006.

[7] Pop, E., Dutton, R., Goodson, K., "Detailed heat generation simulations via the Monte Carlo method", *International Conference on Simulation of Semiconductor Processes and Devices, SISPAD 2003*, 3-5 Sept. 2003.

[8] Rowlette, J., Pop, E., Sinha, S., Panzer, M., Goodson, K., "Thermal phenomena in deeply scaled MOSFETs",*IEEE International Electron Devices Meeting*, IEDM Technical Digest, 5-5 Dec. 2005.

[9] Pop, E., Sinha, S., Goodson, K.E., "Heat Generation and Transport in Nanometer-Scale Transistors",*Proceedings of the IEEE*, vol. 94, Issue 8, pp. 1587 – 1601,

Aug. 2006.

[10] Rowlette, J.A., Goodson, K.E., "Fully Coupled Nonequilibrium Electron–Phonon Transport in Nanometer-Scale Silicon FETs",*IEEE Transactions on Electron Devices*, Volume 55, Issue 1, Jan. 2008 Pages:220 – 232.

[11] Skorek, A.W., Ble, S.V., Gryko-Nikitin, A., Nazarko, J., "Nanothermal Management in Nanoelectronics Systems", *Canadian Conference on Electrical and Computer Engineering, CCECE 2007,* 22-26 April 2007.

[12] Rowlette, J., Pop, E., Sinha, S., Panzer, M., Goodson, K., "Thermal simulation techniques for nanoscale transistors", *IEEE/ACM International Conference on Computer-Aided Design, ICCAD-2005*, 6-10 Nov. 2005.

[13] Gang Chen, "Nanoscale heat transfer and nanostructured thermoelectrics", *IEEE Transactions on Components and Packaging Technologies*, vol. 29, Issue 2, pp. 238-246, June 2006.

[14] Allec, N., Hassan, Z., Shang, L., Dick, R.P.; Yang, R.,"ThermalScope: Multi-scale thermal analysis for nanometer-scale integrated circuits",*IEEE/ACM International Conference on Computer-Aided Design, ICCAD 2008*, 10-13 Nov. 2008.

[15] Ble, S. V., "MODÉLISATION PARALLÈLE DES PHÉNOMÈNES NANOTHERMIQUES", Thesis in Electrical Engineering, University of Quebec in Trois-Rivieres, 2009, p. 120.

[16] GOLDHABER-GORDON D., MONTEMERLO M. S., LOVE J. C., OPITECK G. J., ELLENBOGEN J. C., "Overview of Nanoelectronic Devices", *Proceedings of the IEEE,*vol. 85 , Issue: 4, pp. 521-540, 1997.

[17] ELLENBOGEN J. C., "A Brief Overview of Nanoelectronic Devices",*1998 Government Microelectronics Applications Conference (GOMAC98,)*Arlington, VA, 13-16 March 1998.

[18] Skorek, A.W., "High Performance Computing in nanoscale electrothermal modeling and simulations", *15th International Conference on Mixed Design of Integrated Circuits and Systems, MIXDES 2008*, 19-21 June 2008.

[19] Flynn, M., Some Computer Organizations and Their Effectiveness, *IEEE Trans. Comput.*, vol. C-21, pp. 948, 1972.

[20] www.wikipedia.org

[21] www.computecanada.org

[22] www.clumeq.ca

[23] www.itrs.net

[24] Deleonibus, S. Aid, M. de Salvo, B. Ernst, T. Faynot, O. Fedeli, J.-M. Giffard, B. Le Royer, C. Poiroux, T. Robert, P. Sillon, N. Vinet, M., "New routs and divesifications for nanoelectronics by the end of the roadmap and beyond", *Electron Devices and Solid-State Circuits, EDSSC 2008 IEEE International Conference on*, 8-10 Dec 2008,pp. 1-6.

[25] BehzadRazavi,*Fundamentals of microelectronics*. USA, Wiley, 2008.

[26] L. Dreeskornfeld, J. Hartwich, E. Landgraf, R. J. Luyken, W. Rösner, T. Schulz, M. Städele, D. Schmitt-Landsiedel, L. Risch, "Comparison of partially and fully depleted SOI transistors down to the sub 50nm gate length regime", *Infineon Technologies-Corporate Research Otto-Hahn-Ring*, Institute for Technical Electronics, TU Munich, Germany.

[27] http://www.play-hookey.com/semiconductors/depletion_mode_mosfet.html

[28] ZhibinRen; SebastienGoasguen; Akira Matsudaira; Shaikh S. Ahmed; KurtisCantley; Mark Lundstrom; Xufeng Wang (2006), "NanoMOS," DOI: 10254/nanohub-r1305.11. (DOI: 10254/nanohub-r1305.11).

[29] ZhibinRen, "NANOSCALE MOSFETS: PHYSICS, SIMULATION AND DESIGN", thesis, Purdue University, Octobre 2001.

[30] Xinnan Lin, ChuguangFeng, Shengdong Zhang. Wai-HungHo and iMansun Chan, "Double-Gate SOX MOSFET Fabrication from Bulk Silicon Wafer", *SOI Conference, 2001 IEEE International,* Durango, CO, 2001, pp. 93-94.

[31] Hongmei Wang, Mansun Chan, Singh Jagar, Vincent M. C. Poon, Ming Qin,Yangyuan Wang and Ping K. Ko, "Super Thin-Film Transistor with SOI CMOS Performance Formed by a Novel Grain Enhancement Method", *Electron Devices, IEEE Transactions on*, vol. 47, Issue: 8, pp. 1580 – 1586, 2000.

[32] www.flipchip.com

[33] www.solidworks.com

[34] www.nanohub.org

[35] ZhibinRen, with contributions by: Ramesh Venugopal, Jung-HoonRhew, Jing Guo, Dave Rumsey, Dr. SebastienGoasguen, Prof. SupriyoDatta and Prof. Mark S. Lundstrom, "NanoMOS 2.0 A 2D-Simulator for Double-gate MOSFETs", *Manual*, Purdue University, Decembre 2001.

[36] SebastienGoasguen, "A guided tour of nanoMOS code and some tips on how to parallelize it", *Electron Devices at the Nano/Molecular Scale*, Summer School at UIUC, May 21-22, 2002.

[37] SebastienGoasguen, Ah. R. Butt, Kevin **D.** Colby and Mark S. Lundstrom, "Parallelization of the Nanoscale Device Simulator nanoMOS2.0Using a 100 Nodes Linux Cluster**",** *Proceedings of the 2nd IEEE Conference on Nanotechnology*, IEEE-NANO 2002, November 2002, pp. 409-412.

[38] http://atc.ugr.es/javier-bin/mpitb_eng

[39] www.mathworks.com

[40] Amdahl, G. (April 1967) "The validity of the single processor approach to achieving large-scale computing capabilities". In *Proceedings of AFIPS Spring Joint Computer Conference*, Atlantic City, N.J., AFIPS Press, pp. 483–85.

[41] Gustafson John L., "Reevaluating Amdahl's Law",*Communications of the ACM 31(5)*, 19898, pp. 532–33.