RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ IBN-KHALDOUN DE TIARET

FACULTÉ DES SCIENCES APPLIQUEES
DÉPARTEMENT DE GENIE ELECTRIQUE

MEMOIRE DE FIN D'ETUDES

Pour l'obtention du diplôme de Master

Domaine : Sciences et Technologie

Filière : Electronique

Spécialité : Electronique des Systèmes embarqués

THÈME

Réduction de la consommation énergétique du processeur lors de l'exécution d'un système informatique en temps réel

Préparé par : Sais Djihad Nour Elhouda Asma

Bocoum Ramata Koumbel

Devant le Jury :

Nom et prénoms	Grade	Qualité
D.Nasri	MCA	Président
M.Belarbi	MCA	Examinateur 1
H.Benabid	MAA	Examinateur 2
B.Sahli	MCA	Encadreur

Dédicaces

Ce travail modeste est dédié :

À nos parents Qu'ils trouvent en nous la source de leur fierté à qui nous devons tout ;

et plus particulièrement, à tous nos proches des familles **SAIS** et **BOCOUM**, à tous nos chers amis et nos camarades de l'Université Ibn Khaldoun de Tiaret ; Et à tous ceux qui nous ont enseigné au long de notre vie scolaire.

Remerciement

Tout d'abord, nous remercions le tout puissant ALLAH, notre créateur de nous avoir donné la force, la volonté et le courage afin d'accomplir ce travail modeste.

Nous adressons le grand remerciement à notre encadreur **Mr. SAHLI BELGACEM**, pour ses conseils et sa disponibilité du début à la fin de ce travail, ainsi qu'à l'ensemble des professeurs qui ont contribué à notre formation.

Enfin, nous adressons nos plus sincères remerciements à nos familles qui nous ont soutenues moralement de loin ou de près, toutes les personnes qui ont participé à ces recherches et à l'élaboration de ce mémoire. Ainsi, nos proches et nos amis, qui nous ont accompagnés, aidé, soutenu et encouragé tout au long de la réalisation de ce mémoire.

Sommaire

Chapitre II : Les techniques de la réduction de la puissance

Chapitre III : Les résultats de l'analyse de la puissance

Chapitre IV : Comparaison avec les mesures de la puissance

Liste des abréviations :

CMOS: Complementary Metal Oxyde Semi-conductor

RTL: Register Transfer Level

FPGA: Field Programmable Gate Array

FLPA: Functional Level Power Analysis

ILPA: Instruction Level Power Analysis

DSP: Digital Signal Processor

VLPW: Very Long Instruction Word

CPL: Complex Programmable Logic

DCVSL: Differential Cascade Voltage Switch Logic

DEC: Digital Equipment Corporation

PCB: Printed Circuit Board

CPU: Central Processing Unit

AMD: Advanced Micro devices

FSB: Front-Side Bus

CPI: Cycle Per instruction

MIPS: Millions Instruction Per Second

CISC: Complex Instruction Set Computer

RISC: Reduced Instruction Set Computer

ISA: Instruction Set Architecture

RAM: Random Acess Memory

RMA: Rate Monotonic Analysis

AVX: Advanced Vector Extension

SSE: Steaming SIMID Extension

GPU: Graphics Processing Unit

TDP: Thermal Design Power

SMP: Symmetric Multi-Processing

MOSFET: Metal Oxide Semi-Field Effect Transistor

GSM: Global System for Mobile Communication

PDA: Personal digital Assistant

RTOS: Real Time Operating System

ROM: Read Only Memory

DMA: Direct Memory Acess

EMIF: Extension Memory Interface

HPI: Host Port Interface

MCBSP: Mulichannel Buffered Serial Port

EMIF: Extern Memory Interface

ASV: Adaptive Supply Voltage

SPEC: Standard Performance Evaluation Corporation

ARM: Advanced RISC Machine

HPC-ACE: High Performance Ance Computing, Arithmetic Computational Extensions

FSL: Fujitsu Semiconductor LTD

RTL: Register Transfer Language

IPMI: Intelligent Platform Management Interface

FLOP: Floating Point Operation Per second

PLL: Phase-Locked Loop

RC: Parasitic Resistance & Capacitance

STA: Static Timing Analysis

VCD: Value Change Dump

SI: International System

E/S: Entrées/Sortie

SPARC: Scalable Processor Architecture

IBM: International Business Machine

TLB: Translation Lookaside Buffer

RAPL: Runnig Average Power Limit

Liste des figures

Liste des Tableaux

Introduction Générale :

Les processeurs actuels chauffent beaucoup. Pourtant, ce n'est pas la faute des fabricants de processeurs qui essaient de diminuer la consommation d'énergie de nos CPU au maximum. Malgré cela, nos processeurs voient leur consommation énergétique augmenter toujours plus, l'époque où l'on refroidissait les processeurs avec un simple radiateur est révolue, place aux gros ventilateurs super puissants. Devant tout ce gâchis, on peut légitimement se poser quelques questions : où passe l'énergie engloutie par nos processeurs ?

Les exigences de portabilité des ordinateurs portables et d'autres appareils portatifs imposent de sévères restrictions sur la taille et la consommation d'énergie. Même si la technologie des batteries s'améliore continuellement et que les processeurs et les écrans s'améliorent rapidement en termes de consommation d'énergie, la durée de vie de la batterie et le poids de la batterie influencent considérablement sur la façon dont les ordinateurs portables peuvent être utilisés.

Les dispositifs nécessitent souvent des capacités de traitement en temps réel, et nécessitent donc un débit élevé. La consommation d'énergie devient le facteur limitant de la quantité de fonctionnalités qui peuvent être placées dans ces appareils. Une utilisation plus étendue et continue des services de réseau ne fera qu'aggraver ce problème puisque la communication consomme relativement beaucoup d'énergie. La recherche est nécessaire pour fournir des politiques pour une gestion prudente de la consommation d'énergie tout en fournissant l'apparence de connexions continue aux services et aux applications du système. La distribution d'une horloge à grande vitesse dans un grand système synchrone est à la fois difficile et gourmande en énergie. Il a été suggéré depuis un certain temps que les processeurs asynchrones peuvent donc s'avérer avantageux pour des applications de faible puissance.

La faible consommation d'énergie a été l'un des arguments avancés pour le regain d'intérêt pour la conception asynchrone. L'argument principal est qu'un système asynchrone (ou auto-synchronisé) n'effectue le traitement qu'à la demande. Ainsi, un processeur oisif peut et va s'arrêter. Dans une technologie CMOS à faibles fuites, la logique arrêtée consomme très peu d'énergie.

Cet argument peut être étendu à n'importe quel bloc fonctionnel, de sorte qu'une partie du processeur qui n'est pas requise ne fonctionnera pas et gaspillera de l'énergie. Un exemple de ceci serait le matériel de multiplication dans un processeur à usage général, qui est rarement invoqué. Cet argument est suffisamment convaincant pour que de nombreux processeurs

synchrones de faible puissance utilisent maintenant la commande d'horloge sur leurs sous-composants (circuits partiels).

Une influence plus subtile est que les circuits auto-synchronisés peuvent être conçus pour permettre une variation de la vitesse de fonctionnement dans des circonstances extrêmes. Ceci évite d'avoir à introduire du matériel spécialisé pour accélérer le comportement le plus défavorable et le plus rare dans un cycle d'horloge fixe.

Dans ce mémoire, nous explorerons les sources de consommation d'énergie et fournirons une variété de techniques de réduction d'énergie à différents niveaux du flux de conception d'un système informatique.[1]

CHAPITRE I

- *Analyse du flux de la puissance*

Introduction :

L'analyse de puissance dans la conception de la puce SPARC64 VII a utilisé le taux de fonctionnement moyen en unités de modules de fonction. Cette approche a permis de déterminer la consommation d'énergie pour chaque module de fonction, mais pas pour les cellules individuelles ou les macros. Pour la puce SPAC64 VIII fx qui a été développé en tant que processeur pour l'ordinateur K [1], utilise le procédé CMOS à 45 nm de Fujitsu Semi-conducteur Ltd. pour les semi-conducteurs et est composé de huit cœurs, d'un cache de niveau 2 partagé de 6 Mo et de contrôleurs de mémoire. Nous introduisons un flux d'analyse de puissance au niveau de la porte dans le but d'atteindre une réduction drastique de la consommation d'énergie. Les informations de consommation d'énergie obtenues à partir des résultats de cette analyse de puissance en unités de cellules et macros ont été utilisées dans le processus de conception pour isoler les endroits où des mesures de réduction de la consommation d'énergie étaient nécessaires et pour vérifier leurs effets.

Les informations de débit de fonctionnement utilisées dans cette analyse de puissance en unités de cycles d'horloge utilisant une émulation logique basée sur un cycle sur un émulateur logique. L'information d'exploitation elle-même se composait de taux d'exploitation pour tous les filets et les broches de la cellule à l'intérieur de la puce. De plus, pour conserver la taille du fichier d'informations d'exploitation généré dans cette analyse et le temps nécessaire pour effectuer l'analyse de puissance dans des valeurs réalistes, nous avons sélectionné sur toute la période d'émulation logique plusieurs intervalles d'environ 3000 cycles pour lesquels le taux d'exécution des unités étaient stables à la valeur pointe (peack) et focalisait notre analyse sur ces intervalles pour obtenir des informations détaillées.[2]

I.1 Les transistors (CMOS) :

Vous devez sûrement savoir que nos processeurs sont fabriqués avec des composants électroniques qu’on appelle des transistors, reliés pour former des circuits plus ou moins compliqués. Afin d'expliquer pourquoi notre processeur chauffe et comment diminuer sa consommation énergétique, il va falloir expliquer comment fonctionnent ces transistors utilisés dans nos ordinateurs. Eh oui, la grande partie des pertes énergétiques à l'intérieur d'un processeur a lieu dans ces transistors. De plus, une grande partie des astuces des fabricants repose justement sur les propriétés des transistors utilisés dans nos ordinateurs. Les CMOS consomment moins d’énergie [3].

I.2 Rappels sur le transistor CMOS :

Petite précision pour commencer : il existe différents types de transistors, chacun avec ses particularités, ses avantages et ses inconvénients. On ne va pas en parler plus que ça, mais il faut préciser que les transistors utilisés dans nos ordinateurs sont des transistors à effet de champ à technologie CMOS. Si vous ne comprenez pas ce que ça signifie, ce n'est pas grave, c'est un simple détail. Mais qu'est-ce qu'un transistor CMOS ? Il s'agit simplement d'un composant relié au reste du circuit électronique par trois morceaux de fil conducteur que l'on appelle broches. Nous allons appliquer de force une tension électrique sur ces broches (attention à ne pas la confondre avec le courant électrique), et cette tension représentera soit **0** soit 1 en fonction du transistor utilisé.

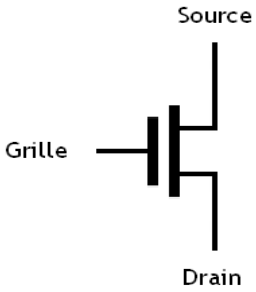


Figure I.1 : Transistor CMOS

Ces trois broches ont des utilités différentes et on leur a donné un nom pour mieux les repérer :

- la grille ;
- le drain ;
- la source.

Dans les processeurs, on utilise notre transistor comme un interrupteur qui réagit en fonction de sa grille : suivant la valeur de la tension qui est appliquée sur la grille, le transistor conduira ou ne conduira pas le courant entre la source et le drain. En clair, appliquez la tension adéquate et la liaison entre la source et le drain se comportera comme un interrupteur fermé et conduira le courant : le transistor sera alors dit dans l'état passant. Par contre, si vous appliquez une autre tension adéquate sur la grille, cette liaison se comportera comme un interrupteur ouvert et le courant ne passera pas : le transistor sera dit dans l'état bloqué. Il existe deux types de transistors CMOS, qui différent entre autres par la tension qu'il faut mettre sur la grille pour les ouvrir/fermer :

- les transistors NMOS qui s'ouvrent lorsqu'on place une tension égale à zéro sur la grille et se ferment si la tension placée sur cette même grille représente un 1 ;
- les PMOS pour qui c'est l'inverse : ils se ferment lorsque la tension sur la grille est nulle, et s'ouvrent si celle-ci représente un 1. [3]

I.3 Anatomie d'un CMOS :

Voyons maintenant ce que notre transistor a dans le ventre. À l'intérieur du transistor, on trouve simplement une plaque en métal reliée à la grille appelée l'armature, un bout de semi-conducteur entre la source et le drain, et un morceau d'isolant entre les deux.

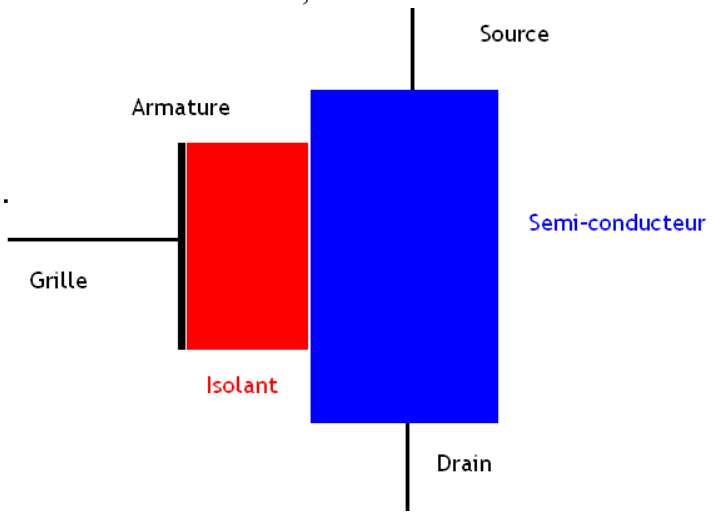


Figure I.2 : Vue générale de l'intérieur du transistor

I.4 Le principe de fonctionnement du CMOS :

Premier cas : on laisse la grille tranquille, la plaque de métal reliée à la grille est vide d'électrons. Si on place une tension entre la source et le drain, un courant électrique va passer et des électrons vont circuler de la source vers le drain : un courant va s'établir. Maintenant, si on place une tension sur la grille, des électrons vont s'accumuler dans la plaque de métal de la grille jusqu'à un certain point. Au bout d'un certain temps, la grille sera remplie. Ces électrons chargés négativement vont donc repousser tous les autres électrons qui circulent entre la source et le drain, ce qui va les gêner et faire circuler ceux-ci plus difficilement : le courant va diminuer.

À partir d'une certaine tension, les électrons stockés dans la grille vont tellement repousser les électrons de la source que ceux-ci ne traverseront plus du tout la liaison entre la source et le drain : le courant ne passe plus. [3]

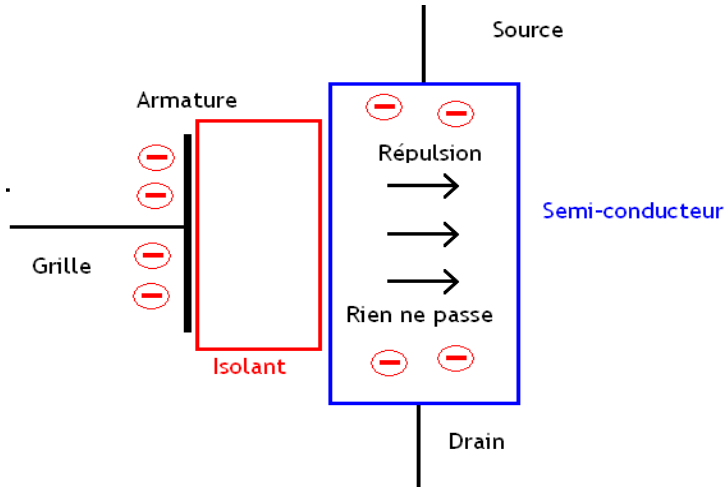


Figure I.3 : Le passage du courant dans un CMOS

I.5 Consommation dynamique :

Il y a plusieurs raisons qui font que nos transistors vont consommer du courant. Nos transistors peuvent consommer quand on les fait changer d'état, ou consommer sans que l'on fasse quoi que ce soit dessus. On doit ainsi faire la différence entre consommation statique et consommation dynamique. Commençons par cette dernière. Cette consommation dynamique correspond à la consommation énergétique du processeur due au fait que nos transistors changent d'état. La première cause d'échauffement dans un transistor est l'accumulation ou la fuite des électrons de la grille lorsque notre transistor change d'état. Pour cela, il faut que la tension de la grille varie de façon à faire passer la plaque de métal de l'état vide à l'état rempli, ou inversement. En vue d'expliquer pourquoi, il faut d'abord remarquer quelque chose : si vous regardez bien, notre transistor est composé de deux morceaux de conducteurs (la grille et la liaison drain-source) séparés par un isolant, c'est une sorte de condensateur. Un condensateur est un composant capable d'accumuler des charges électriques (ici, des électrons). Pour accumuler des charges dans ce condensateur (sur l'armature de la grille), il faut fournir de l'énergie qui sera donc consommée. De même, lorsque l'armature se vide, l'énergie stockée dans cette dernière va se dissiper sous forme de chaleur. L'énergie accumulée par un condensateur est de $12 \times C \times U^2$, avec U la tension appliquée sur l'armature, et C un paramètre du condensateur

nommé sa capacité. Donc, à chaque fois qu'un transistor doit changer d'état, il va pomper une énergie proportionnelle àU^2. [3]

Une partie des pertes énergétiques vient du fait que nos transistors consomment de l'énergie en changeant d'état : on appelle cette perte la consommation dynamique. C'est cette consommation qui intervient pour une grande part dans la consommation énergétique d'un processeur. Les fabricants ont donc cherché un moyen d'estimer celle-ci de façon à pouvoir la diminuer le plus possible. Comme on le voit, suivant la charge de travail que l'on donnera à un processeur, celui-ci chauffera plus ou moins. Difficile donc d'évaluer la quantité de chaleur dissipée et l'énergie consommée sans faire d'approximations. Pour évaluer la quantité de chaleur dissipée en une seconde par notre processeur, on va poser une petite hypothèse : nos transistors n'arrêtent pas de changer d'état et sont exploités au mieux ! Donc, pour un processeur de fréquence f, nos transistors vont changer d'état f fois par seconde.

Pour un seul transistor, l'énergie dissipée en une seconde est donc calculable de la sorte :

$$E = \frac{1}{2} \times C \times U^2 \times f \quad \text{(eq.1)}$$

Pour obtenir la consommation d'énergie totale de notre processeur, il suffit de multiplier celle d'un transistor par le nombre total de transistors de notre processeur (on peut faire ça parce que ces transistors sont tous identiques). En notant le nombre total de transistors du processeur n, on a alors :

$$E_{totale} = \frac{1}{2} \times n \times C \times U^2 \times f \quad \text{(eq.2)}$$

On peut donc en déduire quelque chose : plus un processeur a de transistors, plus celui-ci consommera et chauffera. On peut aussi en déduire qu'un processeur consommera d'autant plus que sa fréquence et/ou sa tension d'alimentation sont élevées.

Vu que n, et C dépendent du processeur et de sa conception, on peut décider de les passer sous le tapis pour obtenir un résultat un peu indépendant de la conception du processeur, en posant :

$$\frac{C \times n}{2} = K \quad \text{(eq.3)}$$

On obtient la formule nous permettant de déduire la consommation d'un processeur en fonction de sa fréquence et de sa tension d'alimentation :

$$E=K \times f \times U^2 \quad \text{(eq.4)}$$

I.6 Consommation statique :

Dans ce qu'ont abordé plus haut, on a vu les solutions et les causes de la consommation dynamique d'un processeur. Cette consommation dynamique correspond à la consommation énergétique du processeur due au fait que nos transistors changent d'état. Mais nous n'avons pas parlé de la consommation statique, celle qui existe même quand nos transistors restent dans le même état. Celle-ci devient de plus en plus importante depuis ces dernières années, et une grande partie (pouvant aller jusqu'à 50 %) de la consommation énergétique d'un processeur passe dans cette consommation dynamique.

Logiquement, la grille et le semi-conducteur d'un transistor sont censés être séparées par un isolant et donc aucun courant ne devrait la traverser. Du moins, c'est la théorie, et la réalité est tout autre : l'isolant est loin de jouer son rôle d'isolant suffisamment bien dans les processeurs actuels. Pourquoi ? À force de vouloir miniaturiser au maximum pour intégrer de nombreux transistors sur le même processeur, la couche d'isolant ne fait guère plus de quelques dizaines atomes d'épaisseur .Inutile de vous dire que celle-ci est loin d'être étanche et que de nombreux électrons quittent la grille pour rejoindre la zone de semi-conducteur, dissipant une partie de l'énergie emmagasinée dans la grille. De plus, ces électrons perdus, il faut alors les remplacer et gaspiller de l'énergie pour remplir à nouveau la grille.

Ces fuites d'électrons à travers l'isolant s'appellent des courants de fuite. Ces pertes par courants de fuite ont lieu en permanence dès que la grille est chargée, que le transistor change d'état ou non. Une faible partie de la consommation énergétique de nos processeurs est due à ces courants de fuite, mais elle devient de moins en moins négligeable au fil du temps, au fur et à mesure que les processeurs diminuent leur consommation dynamique. De nos jours, on estime que 30 à 50 % de la consommation d'un processeur passe dans ces courants de fuite. Autant dire que les diminuer devient assez important. On peut limiter ces courants de fuite en utilisant des matériaux avec un ϵ faible, mais avec une distance grille semi-conducteur si faible, il ne faut pas espérer des miracles. On peut aussi influer sur les courants de fuite en manipulant

la tension d'alimentation. Plus la tension d'alimentation est faible, plus les courants de fuite seront faibles. C'est aussi simple que cela, les deux étant reliés. Nous avons donc une solution pour nos courants de fuite : diminuer la tension d'alimentation de notre circuit. [3]

I.7 Fréquence :

Nos processeurs actuels sont cadencés par ce que l'on appelle une horloge : il s'agit d'une tension cyclique qui sert à synchroniser les circuits présents dans notre processeur. Cette tension change de valeur plusieurs fois par seconde. Le nombre exact de changements de valeur de l'horloge qui ont lieu durant une seconde s'appelle la fréquence du processeur. Cette fréquence se mesure dans une unité : l'hertz.

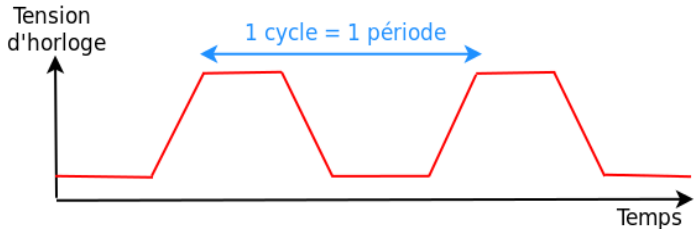


Figure I.4 : Cycle d'horloge

Cette fréquence définit le nombre maximal de changements d'état qu'un transistor peut subir en une seconde : pour une fréquence de 50 hertz, un transistor peut changer d'état 50 fois par seconde maximum. Mais ça ne veut pas dire forcément qu'il le fera : il peut ne pas changer d'état si le circuit dont il fait partie ne fait pas de calculs, il ne dissipera ni ne consommera d'énergie. En effet, dans un circuit qui n'effectue pas de calculs et qui ne fait donc rien, il n'y a aucune raison que les transistors changent d'état. Enfin presque, tout circuit est relié à l'horloge et donc certains transistors du circuit devront tout de même changer d'état, mais nous reviendrons sur cela plus tard. [3]

I.8 Quelques solutions pour réduire la consommation énergétique du processeur :

On voit donc plusieurs solutions simples pour diminuer la consommation énergétique d'un processeur :

- diminuer le nombre de transistors ;
- diminuer la capacité résiduelle de ces transistors ;

- diminuer la fréquence ;
- diminuer la tension d'alimentation du processeur.

On remarque quand même que diminuer la tension d'alimentation est la solution la plus efficace : la consommation énergétique est proportionnelle au carré de la tension, et non à la tension elle-même. Logiquement, les efforts devraient se concentrer sur une diminution de la tension. C'est en pratique ce que font les fabricants de processeurs : ils améliorent leurs transistors de façon à pouvoir diminuer au maximum la tension minimale qui leur permet de fonctionner normalement. En diminuant certains paramètres du transistor, comme la longueur de son armature, l'épaisseur de l'isolant, ou la longueur de la tranche de semi-conducteur, on peut diminuer la tension nécessaire pour faire fonctionner notre transistor.

Et cela, sans toucher à la fréquence. Eh oui, car la fréquence et la tension sont reliées d'une façon qui pose problème quand on souhaite diminuer la tension. Plus la tension d'alimentation d'un transistor est élevée, plus la grille se remplit et se vide vite, plus on peut faire changer d'état le transistor rapidement et donc on peut augmenter la fréquence. Avec les processeurs actuels, la fréquence est devenue tellement démesurée qu'il est difficile de diminuer la tension sans sacrifier en fréquence et donc perdre en performance. Difficile à faire, cependant des solutions techniques existent. Quant au nombre de transistors, nos processeurs actuels embarquent de plus en plus de circuits pour gagner en performance et diminuer le nombre de transistors semble être un vœu pieux, irréalisable. [1]

I.9 Déclenchement de puissance:

Pour limiter les courants de fuite, une autre solution existe. Son principe est simple : diminuer la tension d'alimentation diminue aussi les courants de fuite. Et quand un circuit n'est plus alimenté, ces courants de fuite s'annulent. Alors pourquoi ne pas couper la tension en entrée des circuits inutilisés pour supprimer les courants de fuite ? Eh bien, ceci existe et s'appelle le « power gating » .Cette technique est très efficace, surtout pour couper l'alimentation du cache du processeur. Elle s'implémente en utilisant des transistors, qui se chargeront de déconnecter le circuit de la tension d'alimentation quand ceux-ci sont inutilisés. Ces transistor sont appelés des Sleep Transistors, ou encore des Power Gates. [1]

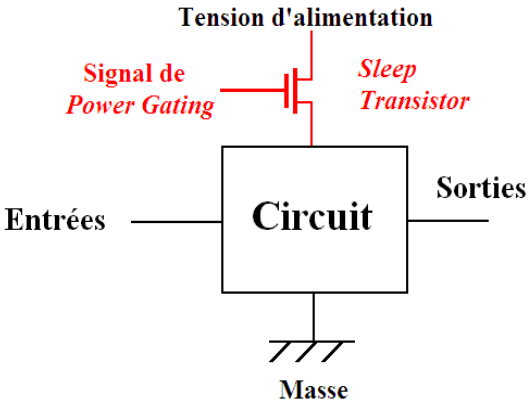


Figure I.5 : Circuit du signal de déclenchement de puissance

I.10 Tensions d'alimentation multiples :

Comme vous l'avez deviné, la fréquence d'horloge du processeur cadence l'intégralité de celui-ci. Elle permet à tous les circuits du processeur d'aller à la même vitesse. Or, certaines portions du processeur sont naturellement plus rapides que d'autres. Il faut dire que certains circuits du processeur sont très simples et donc très rapides, tandis que d'autres sont très complexes et sont plus lents. Si on veut faire aller tous ces circuits à la même vitesse, on doit se limiter au plus petit dénominateur commun, et donc s'aligner sur le circuit le plus lent. Ainsi, certains circuits du processeur sont trop rapides comparés aux autres. Or, vous vous souvenez de la relation entre fréquence et tension. Si ces circuits fonctionnent à une fréquence inférieure à ce qu'ils peuvent, alors pourquoi ne pas baisser leur tension juste ce qu'il faut pour les faire aller à la bonne vitesse ? On économise alors de la tension gratuitement.

Mais dans ce cas, baisser cette tension ne devrait toucher que ces circuits trop rapides. Pour ce faire, on doit utiliser plusieurs tensions d'alimentation pour un seul processeur. Ainsi, certaines portions du processeur seront alimentées par une tension plus faible, tandis que d'autres seront alimentées par des tensions plus élevées. [1]

I.11 Les Flux :

I.11.1 Définition :

Le mot flux désigne en général un ensemble d'éléments (informations / données, énergie, matière…) évoluant dans un sens commun. Un flux peut donc s'entendre comme un déplacement (quelle qu'en soit sa nature) caractérisé par une origine, une destination et un trajet.

I.11.2 Le flux de conception générale et quelques relatifs pour la réduction d'énergie :

Tableau I.1 : La conception générale du flux

Niveau d'abstraction	Exemples
Systèmes	Gestion de la puissance dynamique Méthode de compression Ordonnancement Contrôle d'erreur de communication Protocole d'accès médium Système de mémoire hiérarchique Module spécifique d'application
Logiques	Codage logique Protection des données Gestion d'horloge Logique réversible Conception asynchrone
Technologiques	Réduction de la tension Disposition de la puce Emballage

I.11.3 Les différents types de flux :

I.11.3.1 Le flux lumineux :

I.11.3.1.2 Définition :

Le flux lumineux est la grandeur photométrique qui caractérise la puissance lumineuse d'une source, telle qu'elle est perçue par l'œil humain. Le flux lumineux est le flux énergétique, c'est-à-dire la puissance électromagnétique rayonnée, pondéré par la sensibilité de l'œil humain, normalisée par la fonction d'efficacité lumineuse spectrale, aux différentes longueurs d'ondes. Son unité dans le SI est Lumen (lm).

I.11.3.1.3 Appareil de mesure du flux lumineux (ou de rayonnement) :

On trouve des appareils nommés puissance-mètre ou wattmètre, qui donnent des puissances en watt et donc qui n'ont rien à voir avec la photométrie (qui, elle se cantonne aux lumens émis aux lux-m^2 reçus). Dans le cas d'un wattmètre, il s'agit de watts consommés par le système.

I.11.3.2 Le flux thermique :

I.11.3.2.1 Définition :

Le flux thermique (ou flux de chaleur), souvent noté $\boldsymbol{\Phi}$, entre deux milieux de températures T_i différents correspond au transfert thermique Q, qui s'écoule par unité de temps t entre les deux milieux :

$$\boldsymbol{\Phi}=\frac{\boldsymbol{Q}}{\Delta \boldsymbol{t}} \quad \text{(eq.5)}$$

Ce transfert d'énergie interne est réalisé du corps plus chaud vers le corps le plus froid, ce qui produit une égalisation des températures des deux corps en contact. Le flux thermique s'exprime en J/s ou c'est équivalent en W. Le flux thermique est gouverné par les principes de la thermodynamique et l'équation de la chaleur. Lorsque ce flux thermique traverse une surface S, on obtient une densité de flux de chaleur notée φ telle que :

$$\boldsymbol{\varphi}=\frac{\boldsymbol{\Phi}}{\boldsymbol{S}}. \quad \text{(eq.6)}$$

En régime stationnaire et en l'absence de sources internes de chaleur, la thermiques conserve c'est-à-dire que le flux thermique cédé par le fluide chaud à la paroi égale le flux à travers la paroi, égale le flux cédé par la paroi au fluide froid.

I.11.3.3 Le flux énergétique :

I.11.3.3.1 Définition :

En radiométrie, le flux énergétique ou la puissance rayonnée est la mesure de la puissance totale d'un rayonnement électromagnétique (y compris l'infrarouge, l'ultraviolet et le visible) émise par une source ou reçue par une surface particulière, ou simplement transitant en un point de l'espace. Le flux énergétique est noté $\boldsymbol{\Phi}_e$.

Pour un système physique, le flux énergétique reçu est l'intégrale de l'existence sur toute la surface rayonnante ; tandis que le flux énergétique émis est l'intégrale de la luminance énergétique sur la même surface.

L'unité SI du flux énergétique est le watt (W). Une puissance peut être exprimée en termes d'énergie par unité de temps, soit, toujours en unités SI, en joule par seconde ($J \cdot s^{-1}$ ou J/s).

I.11.3.4 Le flux magnétique :

I.11.3.4.1 Définition :

Le flux magnétique ou flux d'induction magnétique, souvent noté **Φ**, est une grandeur physique mesurable caractérisant l'intensité et la répartition spatiale du champ magnétique. Cette grandeur est égale au flux du champ magnétique noté $\overrightarrow{B}$ à travers une surface orientée$\vec{S}$. Son unité d'expression dans le SI d'unités est le Weber (Wb).

I.11.4 Densité de flux :

I.11.4.1 Définition :

La densité de flux est la quantité de lignes de champ magnétique qui traversent une surface dans un certain point. Son unité dans le SI et donc le watt par mètre carré et par hertz ($\mathrm{W}.m^{-2}.\mathrm{hz}^{-1}$).

CHAPITRE II

- *Les techniques de la réduction de la puissance*

II.1 Introduction :

A l'heure actuelle, la consommation d'énergie est devenue un réel problème dans la conception de dispositifs électroniques dont l'alimentation est assurée par des batteries. L'augmentation des performances et du nombre de fonctionnalités de ces dispositifs est bien plus rapide que l'évolution de la capacité des batteries. En d'autres termes, la demande en énergie est sans cesse croissante tandis que les batteries n'évoluent pas de manière à pouvoir répondre à de tels besoins. Énormément d'appareils souffrent dorénavant de ce problème. Parmi eux, on retrouve les ordinateurs portables, les téléphones portables, les robots mobiles, les satellites, les pacemakers (stimulateur cardiaque). De manière générale, la plupart des dispositifs autonomes en termes d'énergie (appelés << systèmes embarqués>>) sont handicapés par la fracture entre leur consommation et leurs ressources. Ces appareils utilisent des composants électroniques fonctionnant à des fréquences de plus en plus élevées, ce qui garantit une performance sans cesse croissante mais augmente en contrepartie leur consommation en énergie. De la consommation a conduit de nombreux scientifiques à se pencher sur ce problème. Lors de ces vingt dernières années, diverses techniques ont été développées afin que ces équipements puissent assurer leurs performances tout en minimisant leur consommation. Cette réduction de la consommation d'énergie leur permet de s'équiper de batteries plus petites et ainsi de pouvoir eux-mêmes être plus petits et plus légers. De plus, utiliser moins d'énergie permet d'éviter au maximum les risques de surchauffe, ce qui rend ces équipements plus fiables et moins dangereux.

II.2 Les différentes techniques de la réduction de la consommation :

Dans ce paragraphe, nous faisons un tour d'horizon des techniques de réduction de la consommation en énergie, le lecteur pourra se référer à [4] pour un développement plus complet sur ce sujet. Une première stratégie est de travailler sur la technologie des composants matériels. Ainsi une diminution de la taille des composants, rendue possible par des progrès dans les techniques de fabrication, permet une tension d'alimentation plus faible et donc une consommation moindre. Une seconde stratégie est de limiter l'alimentation d'un composant aux blocs nécessaires pour le traitement en cours, par exemple, on peut diviser une mémoire cache en des blocs pouvant être activés indépendamment les uns des autres. Une autre possibilité est de limiter le nombre de changements d'états dans un circuit car chaque changement d'état induit un coût énergétique. Ainsi dans [5], il a été proposé d'utiliser pour

l'adressage mémoire le codage Gray qui garantit un seul bit de différence entre un nombre et son successeur contrairement au classique codage en complément à deux. Il est également possible d'intervenir au niveau de l'architecture matérielle, par exemple en remplaçant un disque dur, gourmand en énergie du fait de son système mécanique, par une mémoire flash. Le dimensionnement de l'architecture matériel est également important. Une autre voie qui est explorée est de spécialiser les composants pour l'usage fait par exemple à l'aide de composants reconfigurables FPGA [6,7]. Enfin, certains travaux étudient l'utilisation de circuits électroniques asynchrones qui, contrairement aux circuits synchrones traditionnels, présentent la caractéristique intéressante de ne consommer de l'énergie que dans les sous-parties du circuit réellement utilisées lors de l'exécution d'une instruction (voir [8] pour un état de l'art).Le logiciel a également un rôle important à jouer pour minimiser la consommation en énergie, par exemple, en optimisant le code des programmes exécutables, ainsi remplacer des opérations de mémoire-à-mémoire par des opérations de registre à registre apportent des gains substantiels [9]. Il peut être aussi parfois bénéfique d'effectuer de l'expansion en ligne (*inlining*) pour limiter le nombre d'appels de fonctions, qui est une opération souvent longue et coûteuse, la contrepartie étant que le code grossit nécessairement et qu'il puisse alors ne plus être contenu dans une mémoire cache.

Il existe naturellement des techniques dites hybrides basées sur une collaboration entre composants matériels et logiciels.

II.3 Les Méthodologies de la réduction de la consommation :

Il est possible d'utiliser des techniques de conception basse consommation à différentes étapes de la conception d'un système. La conception descendante consiste à partir du niveau le plus abstrait d'atteindre le niveau le plus bas. Pour les systèmes, on distingue quatre niveaux:

- le niveau fonctionnel
- le niveau architectural (RTL: Register Transfert Level)
- le niveau logique
- le niveau électrique et physique

Pour chacun des niveaux, les gains sur la consommation, lors de la mise en place de techniques de basse consommation, sont approximativement les suivants (après [10]).

On remarque qu'il est plus intéressant de mettre en œuvre les techniques de basse consommation au début de la démarche de conception qu'à la fin. A tous les niveaux, la

méthode consiste à estimer la consommation des différentes solutions qui sont offertes pour choisir la meilleure. La recherche actuelle se concentre sur les algorithmes d'estimation. Il est à noter que la solution optimale pour la consommation peut être pénalisante du point de vue de la vitesse de traitement et de la taille du circuit (donc de son prix).

II.3.1 Le niveau fonctionnel :

Les méthodologies d'estimation de puissance au niveau système ciblant spécifiquement les processeurs embarqués peuvent être classées en trois catégories : l'analyse de puissance au niveau fonctionnel (en anglais *« Functional Level Power Analysis »* abrégé par FLPA), l'analyse de puissance au niveau instruction (en anglais *« Instruction Level Power Analysis »* abrégé par ILPA) ainsi que les modèles hybrides. L'analyse de puissance au niveau fonctionnel a été introduite dans [14]. La modélisation FLPA, divise le processeur en plusieurs blocs fonctionnels, par exemple l'unité d'extraction (en anglais « *fetch unit* »), l'unité arithmétique logique, la banque de registre entre autres. L'objectif est d'identifier les blocs qui ont un impact sur la consommation de puissance totale du processeur. Après les avoir bien identifiés, chacun de ces derniers doit être décrit individuellement par une expression arithmétique en fonction des paramètres algorithmiques et architecturaux. Les paramètres algorithmiques sont spécifiques à l'algorithme exécuté, comme le taux de *« cache-miss »* et de *« cache-hit »*, tandis que les paramètres architecturaux sont choisis par le concepteur notamment la fréquence d'opération du processeur ainsi comme la longueur des mots. Ensuite, la consommation individuelle de chaque bloc doit être calculée pour plusieurs combinaisons différentes de ces paramètres. Ces différentes combinaisons sont obtenues par l'exécution de plusieurs séquences de code assembleur. Finalement, les données acquises sont analysées afin d'obtenir l'expression arithmétique de chaque bloc fonctionnel.

Au cours des dernières années, la technique FLPA a été étendue et améliorée. Un aspect qui différencie les travaux qui appliquent la FLPA est la séparation appropriée de l'architecture du processeur en blocs fonctionnels. Pour souligner un exemple réel, en [15] la méthodologie FLPA est appliquée pour caractériser un DSP (de l'anglais *« Digital Signal Processor »*) à architecture VLIW (abréviation anglaise pour *« Very Long Instruction Word »)* de Texas Instruments [16], un processeur spécialisé en application multimédia. L'architecture de ce dernier a été divisé en six blocs fonctionnels; l'arbre d'horloge, la cache d'instruction, la cache de données, l'unité arithmétique logique, l'unité d'extraction et la mémoire interne. Également, la méthodologie FLPA est utilisée en [17] pour générer un modèle d'énergie du « soft-processor

» PowerPc405. Le résultat a été intégré dans un modèle System C du processeur permettant l'estimation de puissance à partir d'une simulation de haut niveau. Enfin, basé sur la méthodologie FLPA, SoftExplorer [18] est un outil d'analyse qui inclut une bibliothèque de modèles de puissance couvrant plusieurs blocs fonctionnels d'un processeur. Un profilage rapide du code est nécessaire pour déterminer les paramètres d'entrée des modèles de puissance. Et, une estimation relativement précise se fait dans un délai très court.

II.3.2 Niveaux architectural et logique : chemin de données

II.3.2.1 Réduction de la tension d'alimentation :

La réduction de la tension d'alimentation résulte en une amélioration quadratique de la dissipation de puissance. Par contre, l'effet négatif est une perte de performances temporelles. En effet, diminuer la tension d'alimentation affecte le temps de propagation d'une porte CMOS. Une approximation du premier ordre de ce temps *Td* est donnée ci-dessous :

$$T_d = \frac{\mathrm{Cl}.V_{dd}}{I} = \frac{Cl.V_{dd}}{k(W/L)(V_{dd}-V_t)^2} \qquad \text{(eq.7)}$$

Cl est la capacité de la cellule, *I* est le courant de sortie,$V_t$est la tension de seuil, W et L dépendent de la taille des transistors, et *k* est dépendant du processus technologique. Cette formule devient de plus en plus inexacte avec la tendance submicronique des technologies actuelles. Cependant, la relation inverse entre V_{dd}et $T_d$reste valable.

En observant la relation liant le temps de traversée et la consommation à la tension d'alimentation (voir figure ci-dessous), on peut donc conclure que diminuer cette dernière a un impact important sur la consommation. Il faut faire cependant attention à ne pas trop se rapprocher de la tension de seuil pour éviter d'augmenter de manière exponentielle le temps de propagation. Les résultats montrent qu'il existe une tension d'alimentation optimale se situant autour de 1.5 Volts.

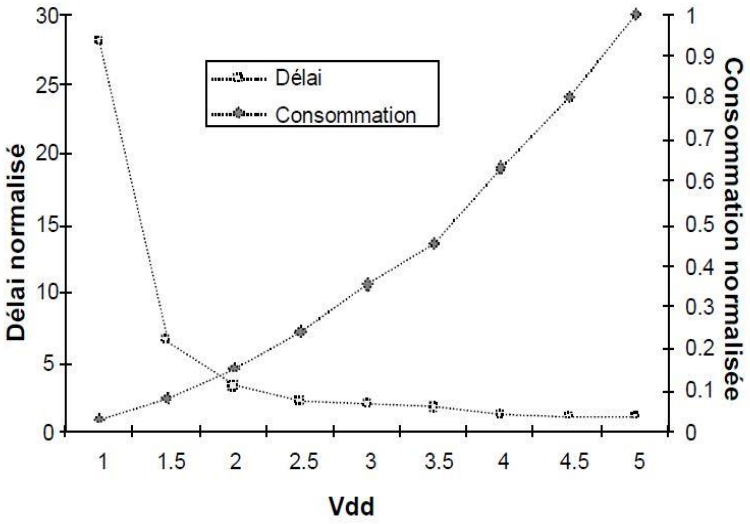
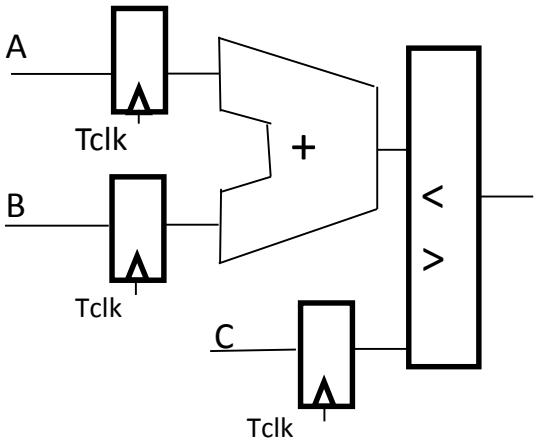
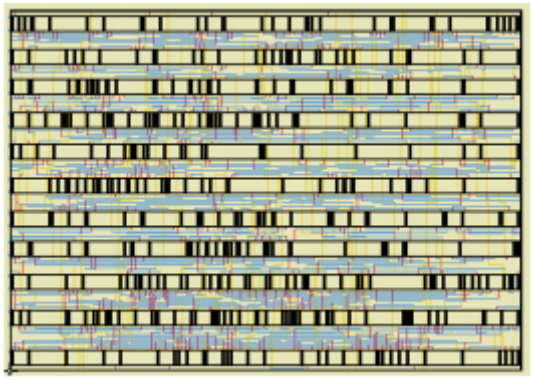


Figure II.1 : Effets de la tension d'alimentation [37]

Cette perte de performances peut être compensée par d'autres moyens aux niveaux logiques ou architecturaux. Par exemple, un additionneur à propagation de retenue **(ripple-carry adder)** peut être remplacé par une structure plus rapide comme un additionneur à anticipation de retenue **(carry lookahead adder).** Ce dernier possède une surface plus importante qui se traduit par une augmentation de la capacité physique. Mais ceci est rattrapé par le fait que l'additionneur le plus rapide peut travailler à une tension d'alimentation plus faible pour des performances équivalentes. D'autres optimisations peuvent être mises en œuvre comme l'utilisation du pipeline ou du parallélisme [13]. L'exemple suivant montre comment des techniques architecturales peuvent être utilisées pour réduire la puissance. La figure ci-dessous consiste en une unité de traitement 16 bits de type addition /comparaison très utilisée dans les systèmes de communication utilisant des codeurs de type Viterbi. Cette structure est issue de [19], et a été calculée en utilisant les outils Compass D.A. et powercalc.

La fréquence de fonctionnement est fixée à 25 MHz, soit un chargement des registres A, B et C avec une période d'horloge de 40ns. On considérera que le chemin critique de ce circuit est de 40 ns, ce qui interdit tout abaissement de l'alimentation.





Fclk=25 MHz

Vdd= 5V

Pmoy = 14.7 mW

Surface= 788*555µ^2= 0.44 mm^2

3823 Transistors

56 mm d'interconnexions

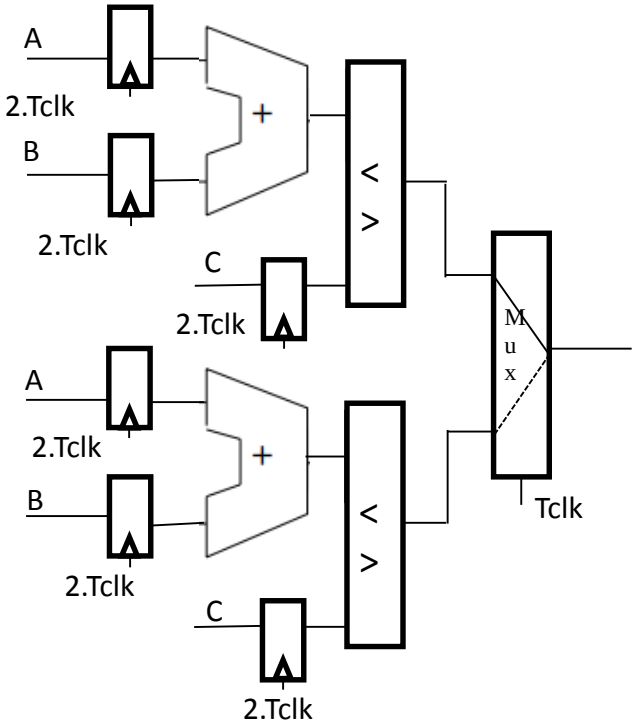
Figure II.2 : Chemin de données Version 1 (schéma et plan tors de masse) [37]

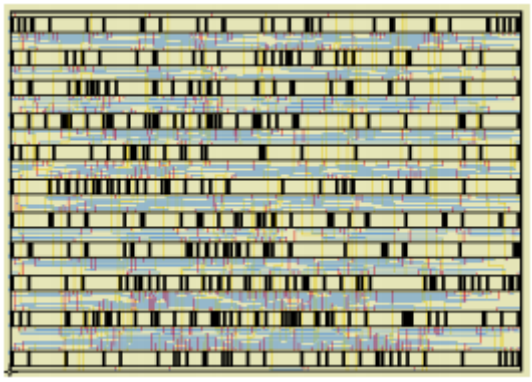
La puissance moyenne consommée par cette UT est donnée par :

$$P_{ref} = C_{ref}.V_{ref}{}^2.F_{ref} = 14{,}7 \text{ mW} \quad \text{(eq.8)}$$

Où C_{ref} est la capacité effective commutée à chaque cycle. Cette capacité a été déterminée par un moyennage de l'activité sur la simulation d'une séquence d'entrée aléatoire de distribution uniforme. $C_{ref} = \alpha . C_{total}$ avec $C_{total} = 31pF$ dans notre cas.

Un des moyens de maintenir les performances tout en abaissant la tension d'alimentation est d'utiliser une architecture parallèle. La figure ci-dessous montre deux unités 16 bits de type addition / comparaison identiques, connectées en parallèle. Chaque unité peut donc travailler à une vitesse moitié par rapport à l'exemple précédent. Les périodes d'horloge des registres passent donc de 40ns à 80ns. L'alimentation peut être abaissée de 5V à 2.9V (utilisation des abaques ou formules de Td en fonction de V_{dd}) pour que le temps du chemin critique du datapath double.





Vdd= 5V

Pmoy=15.4mW

Surface=1130*768µ^2=0.87 mm^2

7638Transistors

111 mm d'interconnexions

Figure II.3 : Chemin de données Version 2 : schéma et plan de masse [37]

La capacité totale de la deuxième version du datapath passe à C_{total}= 64,6 pF, soit un peu plus du double (x2.07). Le bilan de puissance est donc :

$$P_{Version2} = (2{,}15C_{ref}).(0{,}58.V_{ref})^2.(0{,}5.F_{ref}) \# 0{,}36\ P_{ref} = 5{,}3\ mW \quad (eq.9)$$

La puissance moyenne a donc été optimisée par un facteur de 2,8. Le produit puissance x surface a, quant à lui, été divisé par 1,4. Ce type d'optimisation permet donc un gain important en consommation, mais n'est applicable qu'à un système non contraint par la surface. Une autre optimisation possible pour cette UT est d'utiliser le pipeline. Un étage de pipeline peut être introduit entre l'additionneur et le comparateur en diminuant environ le temps du chemin critique de moitié, tout en augmentant la capacité totale de seulement 15%. La tension d'alimentation peut être descendue également jusqu'à 2.9V tandis que l'horloge du système reste identique. Le bilan de puissance devient donc :

$$P_{Pipeline} = (1{,}15C_{ref}).(0{,}58.V_{ref})^2.F_{ref} \# 0{,}39\ P_{ref} \quad (eq.10)$$

II.3.2.2 Réduction de la capacité effective :

Comme la capacité effective est le produit de la capacité physique et de l'activité de commutation, une minimisation de ces deux facteurs doit être envisagée.

II.3.2.2.1 Style logique :

La première approche consiste à utiliser le style de logique minimisant la capacité effective. Dans [19] à l'U.C. Berkeley, on trouvera l'étude du produit puissance x surfaces. D'additionneurs 8 bits implémentés dans différents styles logiques : logique CMOS statique ou dynamique, logique à base de transistors de passage (CPL), logique différentielle (DCVSL). Une comparaison complète de l'étude des différentes logiques peut être trouvée dans la littérature [10].

II.3.2.2.2 Glitch :

Une contribution majeure de consommation de courant est due aux glitch (transitions dynamiques parasites) dans les structures complexes, telles que les additionneurs ou les multiplieurs. Les différents chemins logiques qu'elles contiennent génèrent souvent des transitions parasites sur les signaux internes de ces structures, qui engendrent à leur tour une dissipation supplémentaire de puissance. Les travaux de [21] ont montré qu'une optimisation d'un facteur 5 est possible sur un multiplieur, entre une version carry-save et une version balanced Wallace-tree.

II.3.2.2.3 Activité de commutation :

Le multiplexage de plusieurs opérations sur une même unité peut avoir des effets très néfastes sur la consommation globale d'un système. En effet, cela augmente l'activité de commutation de cette ressource, comme le montre l'exemple d'un bus partagé. La figure ci-dessous montre la comparaison de la consommation de deux compteurs fonctionnant en parallèle [11]. Dans le premier cas, les deux compteurs délivrent leurs résultats sur deux unités de type bus séparées, tandis que dans le second cas, les résultats sont multiplexés sur un seul bus.

L'analyse de l'activité des bus en fonction du décalage entre les états internes des compteurs montre que le premier cas est toujours optimal en consommation si les deux compteurs ne sont pas totalement synchrones.

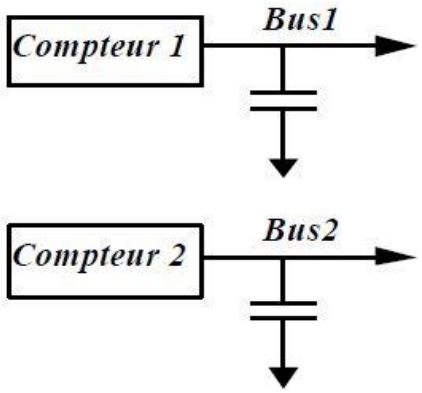


Figure II.4 : Multiplexage temporel

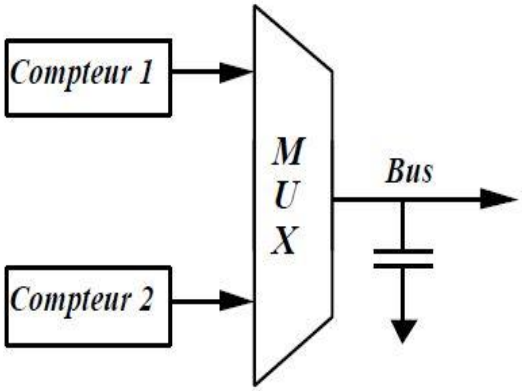


Figure II.5 : Implémentation parallèle

II.3.2.2.4 Multiplexage d'une unité fonctionnelle

En conclusion, lorsqu'on cherche à optimiser la consommation électrique, il ne faut pas utiliser avec excès le partage de ressources. Celui-ci tend à rendre aléatoire la séquence des signaux d'entrée de l'unité fonctionnelle, engendrant ainsi une augmentation de son activité donc de sa capacité effective.

II.3.2.3 Distribution optimisée de l'horloge

La synchronisation des systèmes numériques nécessite un ou plusieurs signaux pour coordonner et séquentialiser les opérations. Le plus souvent, un arbre d'horloge (voir figure ci-dessous) est construit pour fournir le signal d'horloge aux différents modules du système. Ces quelques considérations soulignent l'importance de l'horloge dans les performances du système et de sa consommation totale. En effet, l'énergie dissipée dans un circuit CMOS est toujours dépendante de l'horloge qui gère son activité. Par exemple, le DEC Alpha utilise un générateur d'horloge d'une capacité totale de 3250 Pf, ce qui représente à peu près 40% des 50 Watts dissipés par cette puce.

La construction d'un arbre d'horloge est un point délicat puisqu'elle résulte d'un compromis entre différents objectifs :

- Les horloges étant de plus en plus rapides il faudra ajuster au mieux les longueurs et largeurs des pistes pour réduire les retards (*skew*), ceci entraîne une augmentation des interconnexions et donc de la consommation.

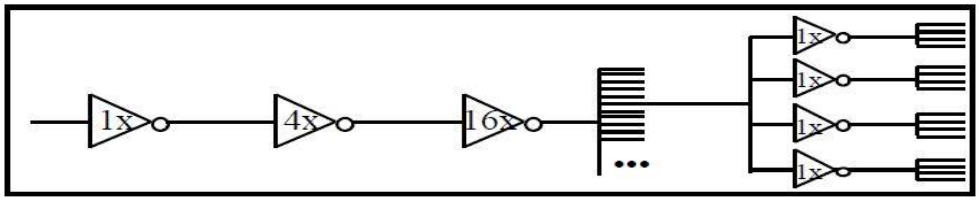


Figure II.6 : Arbre de distribution d'horloge

- Une bufférisassions de type distribué (Figure II.6) améliorera la consommation.
- Diminuer la tension d'alimentation nécessite par exemple d'augmenter la taille des pistes d'horloge pour assurer la même vitesse de transitions des signaux.
- Une distribution au niveau système de l'horloge en évitant de sortir de la puce en passant par un classique *PCB*, c'est à dire en utilisant les technologies *flip-chip* ou *MCM*. Lorsque l'horloge est envoyée sur un bloc fonctionnel, des transitions y sont effectuées même si ce bloc est inutilisé à ce moment: il en résulte une consommation d'énergie abusive. Il est donc intéressant d'inhiber l'horloge des modules inactifs en insérant un module de déconnexion sur le chemin de l'horloge. La figure II.7 présente ce principe sur un contrôleur de type DSP. Le bloc superviseur de l'arrêt des horloges n'est jamais déconnecté.

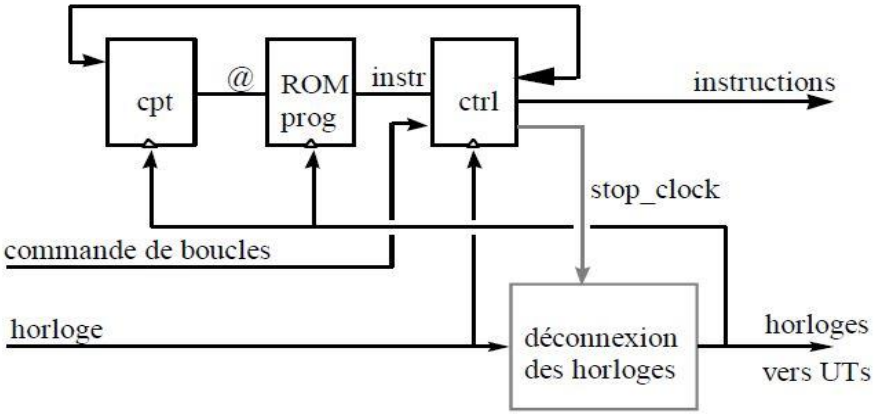


Figure II.7 : Principe de la déconnexion d'horloge

Un problème se pose alors. En effet, la traversée du module de déconnexion de l'horloge ne se fait pas en un temps nul. Donc, lorsque le module est passant, elle arrive avec un temps de retard sur le module de calcul. Ce retard (Skew) peut entraîner des transitions inopportunes (phénomène de Glitch) qui consomment de l'énergie.

D'une façon générale, les circuits asynchrones (c'est à dire sans horloge globale unique) consomment moins à priori. Par contre, leur conception et leur validation est plus difficile [12], [22]. Il parait difficile de concevoir des circuits totalement asynchrones, cependant des techniques mixtes, localement asynchrones [20] sont utilisées.

II.4 Contrainte de temps et d'énergie :

L'apparition de composants électroniques à tension d'alimentation variable constitue un progrès majeur dans l'optique d'une plus grande autonomie et, à l'heure actuelle, de nombreux processeurs comportant cette possibilité sont disponibles commercialement.

On peut citer par exemple les processeurs de la famille Crusoe de la société Transmeta, la technologie PowerNow. D'AMD ou les technologies SpeedStep et XScale d'Intel (pour plus de détails, se référer à [24]). La puissance dynamique dissipée par un composant variant au minimum en le cube de la fréquence de fonctionnement (voir le paragraphe II.5.1), il est judicieux de faire fonctionner le processeur (CPU pour « *Central Processing Unit* » dans la suite) à la fréquence la plus faible compatible avec le niveau de performance requis.

Ainsi, lorsque des contraintes de temps explicites pèsent sur certaines activités du système, il s'agit naturellement de les respecter avec l'objectif supplémentaire de minimiser la consommation en énergie. Le problème d'ordonnancement à résoudre consiste non seulement à déterminer l'ordre dans lequel exécuter les activités du système mais également à fixer la fréquence de fonctionnement du processeur au cours du temps. Comme souligné dans [24], l'ordonnancement sous contrainte d'énergie acquiert une nouvelle dimension qui est la vitesse du processeur. Si aucune contrainte de temps n'est spécifiée alors la meilleure stratégie vis-à-vis de la consommation est de mettre le processeur en veille ce qui naturellement est incompatible avec le niveau de performances minimales attendu. Une technique possible pour garantir le bon fonctionnement du système est alors d'allouer à chacune des activités une date d'échéance, et l'on voit que le problème de l'ordonnancement sous contraintes de temps et d'énergie a des applications en dehors du cadre classique des systèmes temps réel (voir par exemple [25]). Une autre possibilité pour obtenir les performances minimales est de ne pas considérer des échéances individuelles mais de raisonner en termes de nombre minimal de requêtes traitées par unité de temps *(Throughput*) ou de nombre maximum de requêtes en attente de traitement.

II.5 Consommation énergétique d'un processeur :

II.5.1 Historiques sur les processeurs :

II.5.1.1 Définition :

Un processeur (ou CPU, comme central processing unit en anglais) est en quelque sorte le cerveau d'un ordinateur. Il est capable de réaliser des milliards de calculs à la seconde afin d'effectuer n'importe qu'elle opération d'arithmétique.

II.5.1.2 Fonctionnement du processeur :

C'est un circuit électronique cadencé au rythme d'une horloge interne, grâce à un cristal de quartz qui, soumis à un courant électrique, envoie des impulsions, appelées <<Top>>. La fréquence d'horloge appelée également cycle, correspondant au nombre d'impulsions par seconde, s'exprime en Hertz (Hz). Sa vitesse de fonctionnement est calculée en GHz (Plus cette valeur est importante, plus le processeur est puissant). Cependant les deux principaux constructeurs de microprocesseur pour particulier, à savoir Intel et AMD ont choisi une nouvelle stratégie de performances. Ainsi un ordinateur à 200MHz possède une horloge envoyant 200.000.000 de battements. La fréquence d'horloge est généralement un multiple de la fréquence du système (FSB, Front-Side Bus), c'est-à-dire un multiple de la fréquence de la carte mère.

A chaque top d'horloge le processeur exécute une action, correspondant à une instruction ou une partie d'instruction. L'indicateur CPI (cycle par instruction) permet de représenter le nombre moyen de cycles d'horloge nécessaire à l'exécution d'une instruction sur un microprocesseur. La puissance du processeur peut ainsi être caractérisée par le nombre d'instructions qu'il est capable de traiter par seconde. L'unité utilisée est le MIPS (millions d'instructions par seconde) correspondant à la fréquence du processeur que divise le CPI.

II.5.1.3 L'architecture générale du processeur :

II.5.1.3.1 Architecture CISC :

L'architecture CISC (complex instruction set computer, soit <<ordinateur d'instruction complexe>>), consiste à câbler dans le processeur les instructions complexes, difficiles à créer à partir des instructions de base. L'architecture CISC est utilisée en particulier par les

processeurs de type 80x86. Ce type d'architecture possède un coût élevé dus aux fonctions évoluées imprimées sur le silicium.

II.5.1.3.2 Architecture RISC :

Un processeur utilisant la technologie RISC (reduced instruction set computer, soit <<ordinateur à jeu d'instruction réduit>>) n'a pas de fonctions évoluées câblées. Une telle architecture possède un coût de fabrication réduit par rapport aux processeurs CISC.

II.5.1.3.3 Comparaison entre CISC et RISC :

Les processeurs généraux actuels se répartissent en deux grandes catégories appelées CISC pour les instructions complexes et RISC pour les jeux d'instructions réduits. Les processeurs de ces deux catégories se distinguent par la conception de leurs jeux d'instructions. Les processeurs CISC possèdent un jeu étendu d'instructions complexes. Chacune de ces instructions peut effectuer plusieurs opérations élémentaires comme charger une valeur en mémoire, faire une opération arithmétique et ranger le résultat en mémoire. Au contraire, les processeurs RISC possèdent un jeu d'instructions réduit où chaque instruction effectue une seule opération élémentaire. Le jeu d'instructions d'un processeur RISC est plus uniforme. Toutes les instructions sont codées sur la même taille et toutes s'exécute dans le même temps (un cycle d'horloge en général). L'organisation du jeu d'instructions est souvent appelé ISA pour Instruction Set Architecture.

II.5.1.4 : La répartition des principaux processeurs dans les deux catégories :

Tableau II.1 : Les principaux processeurs dans CISC et RISC

CISC	RISC
S/360 (IBM) VAX (DEC) 68xx, 680x0 (Motorola) X86, Pentium (Intel)	Alpha (DEC) PowerPC (Motorola) MIPS PA-RISC(HewlettPackard) SPARC

II.5.2 Puissance de traitement :

Souvent appelée puissance CPU, la puissance de traitement des cycles CPU (qui portent parfois d'autres noms) représente la capacité d'un ordinateur à manipuler des données. La puissance de traitement varie en fonction de l'architecture (et de la vitesse d'horloge) du CPU généralement, des CPU à vitesses d'horloge élevées et des CPU prenant en charge des longueurs de mots plus importantes ont une puissance de traitement supérieure aux CPU plus lents supportant des mots de tailles inférieures. Il est important de garder à l'esprit deux points principaux concernant la puissance de traitement, à savoir :

- La puissance de traitement est fixe
- La puissance de traitement ne peut être stockée

La puissance de traitement est fixe dans le sens où la vitesse du CPU est déterminée. Par exemple, si vous devez ajouter deux nombres (une opération qui, sur la plupart des ordinateurs, ne nécessite qu'une seule instruction pour la machine), un CPU spécifique peut effectuer la tâche à une vitesse déterminée et seulement à cette vitesse spécifique. Mises à part quelques rares exceptions, il n'est même pas possible de ralentir le taux auquel le CPU traite les instructions, alors pour ce qui est de l'augmenter, il ne faut même pas y songer. La puissance de traitement est également fixe à un autre niveau : elle est limitée. C'est à dire qu'il existe des limites quant aux CPU pouvant être branchés dans un ordinateur donné. Certains systèmes sont à même de prendre en charge une vaste gamme de CPU à vitesses variées, alors que d'autres ne pourront peut-être même pas faire l'objet d'une mise à niveau [31].

La puissance de traitement ne peut pas être mise en réserve pour une utilisation ultérieure. En d'autres termes, si un CPU peut traiter 100 millions d'instructions en une seconde, une seconde d'inactivité se traduit par une perte de traitement équivalent à 100 millions d'instructions. En prenant ces informations et en les examinant sous une optique légèrement différente, un CPU peut être considéré comme "produisant" un flux d'instructions exécutées à un taux fixe. Si le CPU "produit" des instructions exécutées, un autre élément doit indubitablement les "consommer". La section suivante se concentre sur ces consommateurs d'instructions.

II.5.2.1 Consommateurs de puissance de traitement :

Les deux principaux consommateurs de puissance de traitement sont :

- Les applications
- Le système d'exploitation lui-même

II.5.2.2 Les applications :

Les consommateurs de puissance de traitement les plus évidents sont les applications et les programmes que vous souhaitez voir exécuter par l'ordinateur. Qu'il s'agisse d'un tableur ou d'une base de données, ces outils sont les raisons pour lesquelles vous avez un ordinateur. Un système à CPU unique ne peut effectuer qu'une seule tâche à un moment donné. Par conséquent, si votre application est en cours d'exécution, aucun autre élément de votre système ne peut être exécuté. Et bien évidemment, il en va de même pour la relation inverse si tout élément autre que votre application est en cours d'exécution, cette dernière ne pourra être exécutée.

Dans de telles conditions, comment se fait-il que de nombreuses applications différentes puissent en apparence tourner sous un système d'exploitation moderne ? La réponse est simple ; ces systèmes d'exploitation sont des systèmes multitâches. En d'autres termes, ils créent l'illusion que de nombreux éléments sont exécutés simultanément alors que c'est en fait impossible. L'astuce consiste à donner à chaque processus une durée d'exécution sur le CPU d'une fraction de seconde avant d'accorder le même traitement à un autre processus qui lui, disposera de la fraction de seconde suivante. À condition que ces changements de contexte se produisent suffisamment rapidement, il est possible de créer l'illusion que de multiples applications sont exécutées de manière simultanée.

Bien sûr, les applications effectuent des tâches autres que la seule manipulation de données par le biais du CPU. Elles peuvent aussi bien attendre des entrées de l'utilisateur qu'effectuer des activités d'E/S vers des périphériques comme les disques durs ou des affichages de graphiques. Lorsque ces événements se produisent, l'application n'a plus besoin du CPU. Dans ces cas-là, le CPU peut être utilisé pour d'autres processus exécutant d'autres applications sans pour autant ralentir du tout l'application en attente.

En outre, le CPU peut être utilisé par un autre consommateur de puissance de traitement : le système d'exploitation lui-même.

II.5.2.3 Le système d'exploitation :

Il est difficile de déterminer le degré de puissance de traitement consommé par le système d'exploitation. En effet, pour effectuer leur travail, les systèmes d'exploitation utilisent une combinaison de codes au niveau des processus et au niveau du système. Alors qu'il est facile d'utiliser par exemple, un contrôleur de processus pour déterminer ce que fait le processus exécutant un démon ou service, il est plus difficile de déterminer la puissante de traitement consommée par le traitement d'activités en relation avec les E/S au niveau du système (ce qui est normalement effectué dans le contexte du processus demandant les E/S).

En général, il est possible de diviser ce genre de temps de gestion du système d'exploitation en deux types :

- Gestion interne du système d'exploitation
- Activités en relation avec les processus

La gestion interne du système d'exploitation inclut des activités telles que la programmation des processus et la gestion de la mémoire alors que les activités associées aux processus incluent tout processus supportant le système d'exploitation lui-même, tels que des processus traitant la journalisation des événements dans tout le système ou effectuant le vidage du cache des E/S (cache flashing).

II.5.3 Réponse à une insuffisance de processeur :

Lorsque la puissance de traitement disponible est insuffisante pour faire face au travail devant être effectué, deux options sont disponibles :

- Réduction de la charge
- Augmentation de la capacité

II.5.3.1 Réduction de la charge :

La réduction de la charge CPU est une opération ne nécessitant aucun coût supplémentaire. L'astuce consiste à identifier les aspects de la charge du système qui sont sous votre contrôle et qui peuvent être réduits. À cet égard, trois aspects ont une importance particulière :

- La réduction du temps de gestion du système d'exploitation
- La réduction du temps de gestion des applications
- L'élimination totale des applications

II.5.3.2 La réduction du temps de gestion du système d'exploitation :

Afin de réduire le temps de gestion du système d'exploitation, il est nécessaire d'examiner la charge actuelle de votre système afin d'identifier les aspects précis entraînant des temps de gestion excessifs. Parmi ces derniers pourraient figurer :

- La réduction du besoin de programmation fréquente de processus
- La réduction de la quantité d'E/S effectuées

Ceci dit, il ne faut pas s'attendre à des miracles ; dans un système raisonnablement bien configuré, il y a peu de chances que la seule réduction du temps de gestion du système d'exploitation entraîne une amélioration considérable de la performance. En effet, un système raisonnablement bien configuré n'a, par définition, qu'un temps de gestion minimal. Toutefois, si votre système tourne par exemple avec une quantité de RAM insuffisante, vous serez peut-être en mesure de réduire le temps de gestion en résolvant le problème d'une mémoire vive trop petite.

II.5.3.3 Réduction du temps de gestion des applications :

Pour réduire le temps de gestion d'une application, il est nécessaire de s'assurer que cette dernière dispose bien de tout ce dont elle a besoin pour fonctionner correctement. Certaines applications ont des comportements extrêmement différents selon les environnements une application peut par exemple voir ses capacités fortement limitées par le traitement de certains types de données, mais pas par le traitement d'autres.

À ce stade, il est important de garder à l'esprit qu'une bonne compréhension des applications exécutées sur votre système est nécessaire pour pouvoir leur permettre de tourner aussi efficacement que possible. Pour ce faire, vous devrez souvent travailler avec vos utilisateurs et/ou les développeurs de votre entreprise afin de pouvoir découvrir des moyens permettant aux applications de tourner plus efficacement.

II.5.3.3.1 Élimination complète des applications :

Selon l'environnement de votre entreprise, il est possible que cette approche ne puisse pas être utilisée, car il n'est souvent pas du ressort de l'administrateur système d'imposer une liste des applications qui peuvent ou ne peuvent pas tourner. Toutefois, si vous pouvez identifier des applications qui sont connues pour leur "monopole de CPU", vous serez peut-être en mesure d'influencer les personnes responsables de ce genre de décision, afin que les applications en question soient supprimées.

À cet égard, vous ne serez vraisemblablement pas la seule personne impliquée. Les utilisateurs concernés devraient certainement faire partie de ce processus ; dans bien des cas, ils disposeront probablement des connaissances et du pouvoir politique nécessaires pour apporter les changements nécessaires à la liste des applications.

II.5.4 Augmentation de la capacité :

Bien sûr, s'il n'est pas possible de réduire la demande de puissance de traitement, vous devez trouver des moyens d'augmenter la puissance de traitement disponible. Cette option est certes tout à fait possible, mais elle entraînera des dépenses.

II.5.4.1 Mise à niveau du CPU :

L'approche la plus simple consiste à déterminer s'il est possible de mettre à niveau le CPU de votre système. Pour ce faire, la première étape consiste à voir si le CPU actuel peut être retiré. Certains systèmes (essentiellement les ordinateurs portables) dotés de CPU soudés ne permettent pas une mise à niveau. Sur les autres en revanche, les CPU sont branchés et rendent donc les mises à niveau possibles tout du moins en théorie. Il convient ensuite de faire un peu de recherche afin de déterminer si un CPU plus rapide existe pour la configuration de votre système. Par exemple, si vous disposez actuellement d'un CPU de 1GHz et qu'il existe une unité de 2GHz du même type, une mise à jour sera peut-être possible.

Finalement, il est essentiel de déterminer la vitesse d'horloge maximale prise en charge par votre système. Toujours sur la base de l'exemple précédent, même s'il existe un CPU de 2GHz d'un type approprié, un simple échange de CPU n'est pas vraiment une option si votre système ne prend en charge que des processeurs tournant à une vitesse égale ou inférieure à 1GHz. Dans le cas où vous ne pourriez pas installer un CPU plus rapide dans votre système, l'étendue de vos

options se limitera peut-être au remplacement des cartes mères ou à la mise à niveau massive dont nous avons parlé précédemment.

Toutefois, certaines configurations système permettent une approche légèrement différente. Au lieu de remplacer le CPU actuel, pourquoi ne pas tout simplement en ajouter un autre ?

II.5.5 Le traitement multitâche symétrique : (SMP)

Le traitement multitâche symétrique (aussi appelé multitraitement symétrique ou SMP de l'anglais Symmetric multi-processing) permet à un ordinateur d'avoir plus d'un CPU partageant toutes les ressources systèmes. Ainsi, contrairement à un système doté d'un seul processeur, un système SMP peut, lui, faire tourner plus d'un processus à la fois. Au premier abord, cette situation semble représenter le rêve d'un administrateur de système. SMP permet avant tout d'augmenter la puissance de CPU d'un système, même s'il existe des CPU avec une vitesse d'horloge plus rapide simplement en ajoutant un autre CPU. Cette souplesse s'accompagne néanmoins d'un certain nombre de contraintes.

La première d'entre elles est que tous les systèmes ne sont pas en mesure de fonctionner en traitement multitâche symétrique (SMP). La carte mère de votre système doit être conçue pour prendre en charge de multiples processeurs. Si ce n'est pas le cas, la mise à niveau de la carte mère (au strict minimum) sera nécessaire.

Le deuxième d'entre elles est que SMP augmente le temps de gestion du système. En y réfléchissant bien, l'augmentation du nombre de CPU pour lesquels la programmation de tâches doit être effectuée se traduit logiquement, pour le système d'exploitation, en un plus grand nombre de cycles CPU au niveau de la gestion. En outre, un plus grand nombre de CPU peut entraîner une contention accrue pour les ressources système. En raison de ces facteurs, la mise à niveau d'un système à double processeur vers une unité à quadruple processeur n'entraîne pas une augmentation à 100% de la puissance CPU disponible. En fait, en fonction du matériel même, de la charge de travail et de l'architecture du processeur, il est possible d'arriver à un stade où l'ajout d'un autre processeur pourrait en fait réduire la performance du système.

Il est aussi important de se souvenir que SMP ne permet pas d'influencer des charges de travail composées d'une application monolithique avec un seul flux d'exécution. En d'autres termes, si un grand programme de simulation nécessitant une activité de calcul intense tourne en tant qu'un processus et n'a pas de fils (ou thread), il ne sera pas exécuté plus rapidement sur un système SMP que sur un ordinateur à processeur unique. En fait, il tournera même peut-être

un peu plus lentement à cause de la gestion supplémentaire engendrée par SMP. Ainsi, de nombreux administrateurs de système estiment qu'en matière de CPU, une puissance de traitement à flux unique est l'option à retenir. Elle offre une puissance CPU optimale avec des restrictions minimales au niveau de son utilisation.

Même si ces informations semblent indiquer que SMP n'est jamais une bonne option, il existe certaines situations dans lesquelles un tel choix est tout à fait approprié. Des environnements faisant tourner de multiples applications à calculs intensifs représentent, par exemple, un environnement tout à fait approprié pour SMP. En effet, des applications dont la tâche n'est autre que d'effectuer des calculs pendant de longues périodes de temps maintiennent la contention entre les processus actifs (et par conséquent, le temps de gestion du système d'exploitation) à un minimum, alors que les processus eux-mêmes utilisent chaque CPU. Il est important de se rappeler également que la performance d'un système SMP à tendance à se dégrader plus progressivement au fur et à mesure que la charge du système augmente. C'est précisément une des raisons pour lesquelles les systèmes SMP sont populaires dans des environnements de serveurs ou dans des environnements à utilisateurs multiples, car l'impact qu'un mélange de processus changeant constamment a sur la charge du système entier est moindre sur un ordinateur à processeurs multiples.

II.5.6 Puissance dissipée et consommation énergétique :

Les concepts de cette section sont valables pour tout circuit CMOS (**Complementary Metal Oxide Semiconductor**) qui est la technologie dominante dans les circuits électroniques. Le lecteur désirant des développements plus approfondis sur ce sujet pourra consulter [24]. L'énergie consommée dans un intervalle de temps [a,b] est par définition l'intégrale de la puissance dissipée $E = \int_a^b P(t)dt$ où P(t) est la puissance dissipée à l'instant *t*. Cette puissance dissipée dans un circuit électronique se compose de la puissance statique et de la puissance dynamique. Dans les circuits CMOS la puissance dynamique représente de l'ordre de 80-85 % de la puissance dissipée et, classiquement, on néglige la puissance statique1. La puissance dissipée totale peut donc s'exprimer par :

$$P \approx P_{dynamique} \sim \alpha\, f\, C V^2 \quad \text{(eq.11)}$$

Où α est le nombre de transitions par cycle d'horloge, f est la fréquence de fonctionnement. *C* est la capacité équivalente et V est la tension d'alimentation. Le terme α dépend des données

traitées et de la technique de codage utilisée, C'est une caractéristique du circuit utilisé. Réduire la fréquence f sans modifier la tension sera sans effet au niveau de la consommation car, globalement, le temps nécessaire pour terminer une même séquence de code augmente d'un facteur k si l'on réduit la fréquence d'un même facteur *k*. Il est finalement possible de diminuer la tension V mais fréquence et voltage sont liés par la relation :

$$\frac{1}{f} \sim \frac{V}{(V - V_t)^{\gamma}} \quad \text{(eq.12)}$$

Avec V_t la tension de seuil et γune constante. Pour une tension de seuil suffisamment petite par rapport à la tension d'alimentation, la relation entre fréquence et tension d'alimentation devient f $\sim V^{\gamma-1}$. Dans le modèle MOSFET (**Metal Oxide Semiconductor Field Effect Transistor**) classique, γ est approximé par deux ; la fréquence est donc linéaire en la tension et la puissance varie en le cube de la fréquence. Certains autres modèles considèrent des valeurs différentes de γ(par exemple γ = 1,3) dans [23] cité dans [24]) mais, en pratique, il n'est pas crucial de déterminer l'expression exacte de la puissance car la puissance dynamique dissipée reste toujours une fonction convexe croissante de la fréquence et beaucoup de résultats énoncés en économie d'énergie sont valables pour toute fonction convexe croissante. Pour une tension d'alimentation donnée, il existe une fréquence de fonctionnement optimale du point de vue énergétique qui est la fréquence maximale supportée par le circuit à cette tension. Dans la suite, plutôt que de raisonner en termes de fréquence, nous parlerons de la vitesse du processeur qui est le rapport entre la fréquence de fonctionnement courante et la fréquence maximale du processeur, appelée aussi fréquence nominale.

Tableau II.2 : Consommation des différents processeurs

Processeur	Horloge (MHz)	Technologie (µm)	Alimentation (V)	Consommation Maximale (W)
Intel Pentium	66	0.8	5.0	16
Intel P6	200	0.35	3.3	35
DEC Alpha 21064	200	0.75	3.3	30
DEC Alpha 21164	300	0.5	3.3	50
PowerPC 620	133	0.5	3.3	30
MIPS R10000	200	0.5	3.3	30
UltraSparc	167	0.45	3.3	30
PowerPC 603	80	0.5	3.3	2.2
IBM 486SLC2	66	0.8	3.3	1.8
MIPS R4200	80	0.64	3.3	1.8

II.6 Les deux formes d'autonomie :

Le moyen le plus simple de définir un système embarqué pourrait être << système qui n'est pas relié physiquement à dispositif fixe >>, mais si cette définition a le mérite d'être simple, elle ne permet pas d'en connaître directement les caractéristiques. C'est pour cela qu'il est plus judicieux de définir un système embarqué comme une unité de traitement possédant une certaine autonomie. Cette autonomie est la propriété la plus couramment associée à la notion de système embarqué et elle peut se décliner sous plusieurs formes.

II.6.1 Autonomie de fonctionnement :

Afin de pouvoir fonctionner de manière autonome, le système embarqué doit disposer de l'ensemble minimum des éléments physiques d'un système : processeurs, mémoire, entrées/sorties, ainsi que d'un programme contrôlant ces différents éléments. Il doit également disposer d'une source d'énergie : générateur ou batteries, la nature de cette source pouvant influer sur le comportement du système, avec, par exemple, une gestion de la consommation dans le cas des batteries.

Pour étendre l'autonomie de fonctionnement d'un système, seules les deux méthodes existent : augmenter la quantité d'énergie embarquée ou diminuer la consommation du système.

La première solution a entraîné de nombreuses recherches dans le domaine des batteries, mais malgré les progrès effectués dans ce domaine, il est toujours difficile d'augmenter la capacité d'une batterie sans en augmenter le poids, le volume et le prix. La seconde solution est complémentaire à la première, et a également donné lieu à de nombreuses recherches dans le domaine de l'électronique et de l'informatique.

II.6.2 Autonomie fonctionnelle :

La différence entre autonomie de fonctionnement et autonomie fonctionnelle apparaît au niveau des services. Un système embarqué offrant une autonomie fonctionnelle est capable de fournir ses services sans aucun recours à des composants qui lui sont externes, alors qu'un système embarqué sans autonomie fonctionnelle dépend toujours de structures extérieures.

Un téléphone portable, par exemple, peut fonctionner parfaitement (système complet et batteries pleines), mais ne peut pas être capable de fournir le service de communication attendu s'il se trouve hors zone de réception des cellules GSM. Il s'agit donc d'un système autonome du point de vue du fonctionnement mais pas du point de vue des fonctionnalités. En revanche, une calculatrice de poche présente une autonomie fonctionnelle évidente.

Un système peut également présenter une certaine autonomie fonctionnelle, mais offrir des services plus étendus lorsqu'il se trouve dans un environnement adéquat. Un cas typique est celui de l'ordinateur portable équipé d'un modem, il possède une autonomie fonctionnelle, mais on peut le relier à un l'internet ou à un serveur distant à l'aide d'un réseau filaire ou cellulaire afin de profiter de nouveaux services offerts par des prestataires distants.

II.7 Le temps réel :

II.7.1 Définition :

Un système est dit temps réel si l'exactitude des applications ne dépend pas seulement du résultat, mais aussi du temps auquel ce résultat est produit. Un système temps réel est un système informatique soumis à des contraintes de temps.

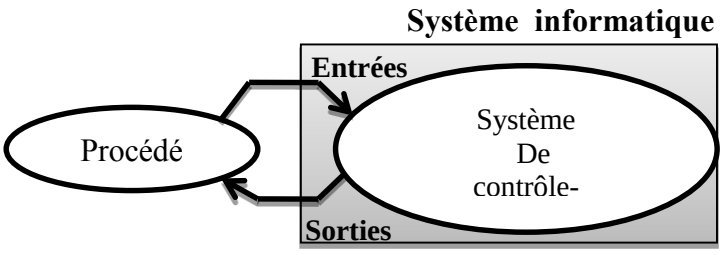


Figure II.8 : Système temps réel

Beaucoup de systèmes embarqués interagissent directement avec leur environnement, afin d'obtenir des valeurs d'entrées (capteurs) ou bien pour appliquer le résultat d'un traitement (actionneur). Mais certains ont des contraintes temporelles nettement plus fortes que d'autres. Qu'un PDA mette une ou trois secondes à afficher l'adresse que l'on recherche dans son annuaire, cela n'est pas un problème quand l'information est toujours utilisable malgré les deux secondes de retard. Mais si un téléphone portable a plusieurs dizaines de millisecondes de retard dans le décodage des trames GSM, la conversation deviendra incompréhensible pour l'utilisateur. Les systèmes embarqués peuvent donc être confrontés à des environnements logiciels extrêmement hétérogènes pouvant aussi bien inclure des contraintes temporelles fortes qu'une absence de totale de notion temporelle.

Deux techniques sont utilisées pour satisfaire les contraintes temporelles : les développements monolithiques et les RTOS. Les développements monolithiques sont effectués dans un langage de bas niveau, ils consistent à écrire un seul programme qui aura la charge du fonctionnement du système dans son intégralité. Le respect des contraintes temporelles est obtenu par la conception même du programme. A l'inverse, les RTOS fournissent aux développeurs des mécanismes adaptés à la programmation temps-réel comme des ordonnanceurs à priorités, l'héritage des priorités **(SRL90)** lors des blocages sur des ressources partagées, et des mécanismes de communications en temps borné. Ces RTOS permettent l'utilisation de techniques plus évoluées, comme l'analyse RMA **(LL73)** pour le développement des applications temps-réels.

II.7.2 Type de systèmes temps-réel :

Il existe deux :

-**Temps-réel strict/dur (hard real-time) :** le non-respect d'une contrainte de temps a des conséquences graves (humaines, économiques, écologiques) : besoin de garanties

-**Temps-réel souple/mou (soft real-time) :** on peut tolérer le non-respect occasionnel d'une contrainte de temps (garanties probabilistes).

II.8 Les lois de Moore et Gilder :

II.8.1 Loi de Moore :

Gordon Earle Moore est un cofondateur avec Robert Noyce et Andrew Grove de la société Intel en 1968. Il avait affirmé dès 1965 que le nombre de transistors par circuit de même taille allait doubler, à prix constant, tous les ans. Il rectifia par la suite en portant à dix-huit mois le rythme de doublement. Cette loi de Moore se vérifie encore aujourd'hui [32].

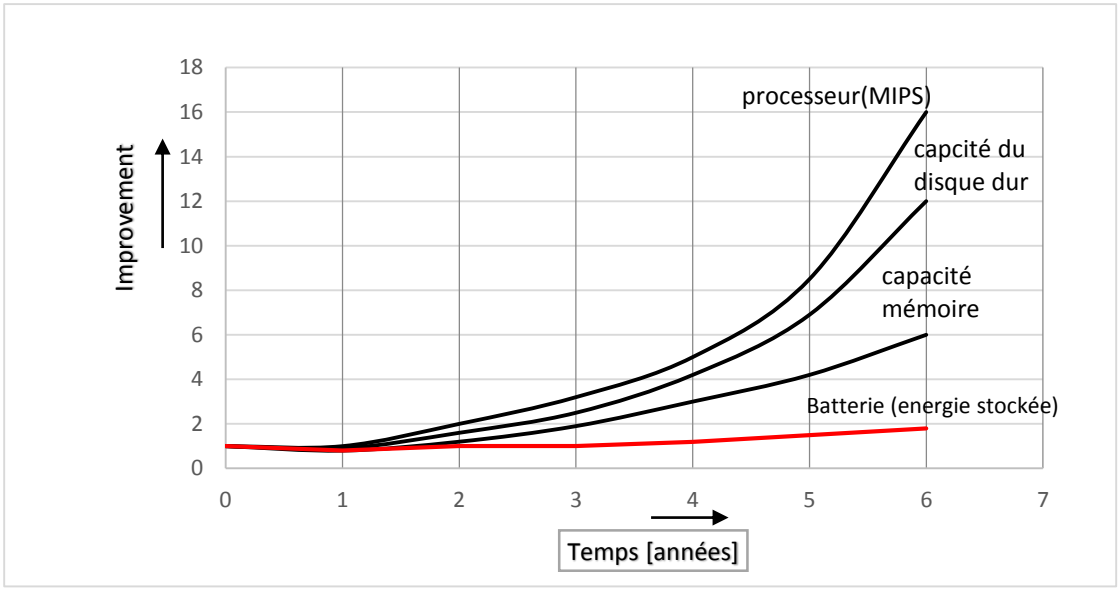


Figure II.9 : La courbe de Moore[32]

II.8.2 Loi de Gilder :

George Gilder, né en 1939 à New-York, est un scientifique utopiste. Il est l'auteur visionnaire de Telecosm, mais aussi le rédacteur du Gilder Report. La bande passante croît au moins trois fois plus rapidement que la puissance des ordinateurs.

Si on prend en compte la loi de Moore qui stipule que la puissance des ordinateurs double tous les dix-huit mois, alors la largeur de bande double tous les six mois [33].

II.9 Traitement numérique du signal : (DSP)

II.9.1 Définition :

Un DSP (Digital Signal processor) est un type particulier de microprocesseur. Il se caractérise par le fait qu'il intègre un ensemble de fonctions spéciales. Ces fonctions sont destinées à le rendre performent particulièrement dans le domaine du traitement numérique du signal.

Comme un microprocesseur classique, un DSP est mis en œuvre en lui associant de la mémoire (RAM, ROM) et des périphériques. Ce DSP typique a plutôt vocation à servir dans des systèmes de traitements autonomes.

Il se présente donc généralement sous la forme d'un microcontrôleur intégrant, selon les masques et les gammes des constructeurs, de la mémoire, de timers, des ports série synchrones rapides, des contrôleurs DMA, des ports d'E/S divers [34].

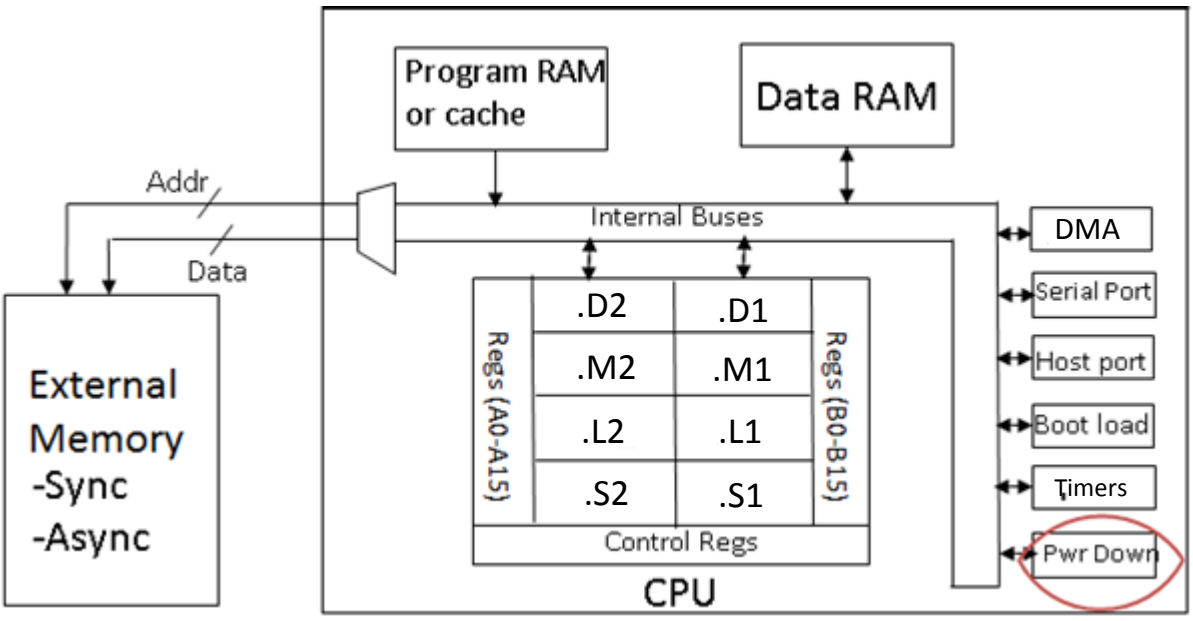


Figure II.10 : Schéma du TMS320C6X

Le CPU du C6x contient huit unités fonctionnelles (Figure II.10 **)** divisé en deux parties. Chaque unité à ce qu'on appelle :

.M : utilisée pour la multiplication.

.L : utilisée pour les opérations logiques et arithmétiques.

.S : utilisée pour les branchements, manipulation de bit et opérations arithmétiques.

.D : utilisée pour le chargement, le stockage et les opérations arithmétiques. Certaines instructions telle que ADD sont effectuées par plusieurs unités.

Les bus internes sont constitués d'un bus d'adresses de programme de 32 bits, d'un bus de données programme de 256 bits acceptant huit instructions 32 bits, de deux bus d'adresses de données 32 bits (DA1 et DA2), bits (64 bits pour la version C64), bus de données de chargement (LD1 et LD2) et deux bus de données de stockage 32 bits (64 bits pour la version à virgule flottante) (ST1 et ST2). De plus, il existe des données d'accès direct à la mémoire (DMA : Direct Memory Acess) de 32 bits et un bus d'adresse DMA de 32 bits. La mémoire hors puce ou externe est accessible via un bus d'adresses de 20 bits et un bus de données de 32 bits. Les périphériques sur un processeur C6x typique incluent l'interface de mémoire externe (EMIF :Extern Memory Interface), le DMA, le chargeur d'amorçage, le port série multicanaux tamponné (McBSP), l'interface de port d'hôte (HPI), le temporisateur et l'unité de mise hors tension (Pwr Down) qui n'existait pas au pare avant dans les architectures mais actuellement elle existe et son rôle est surtout pour économiser l'énergie si le système oisif. EMIF fournit le timing nécessaire pour accéder à la mémoire externe. DMA permet le mouvement des données d'un endroit dans la mémoire à un autre endroit sans interférer avec le fonctionnement de l'unité centrale. Boot Loader démarre le code depuis la mémoire hors puce ou HPI vers la mémoire interne. McBSP fournit une liaison de communication série multicanal à haute vitesse. HPI permet à un hôte d'accéder à la mémoire interne. La minuterie fournit deux compteurs 32 bits. L'unité de mise hors tension est utilisée pour économiser de l'énergie pendant des durées lorsque le processeur est inactif. En général, il existe trois étapes de base pour exécuter une instruction. Ils incluent la récupération, le décodage et l'exécution. Si ces étapes sont effectuées en série, toutes les ressources du processeur, telles que les bus multiples ou les unités fonctionnelles, ne sont pas entièrement utilisées. Afin d'augmenter le débit, les processeurs DSP sont conçus pour être pipelinés.

II.10 Les potentiels énergétiques :

Avec l'augmentation des fonctions de calcul et de communication souhaitées pour les systèmes mobiles sans fil, la densité d'énergie des technologies de batteries existantes est loin de ce qui est nécessaire. Le tableau 4 montre les potentiels énergétiques de la technologie des batteries actuelles.

Les progrès les plus récents dans le domaine des batteries d'ordinateurs portables se traduisent par une meilleure «jauge de carburant» de la batterie, pour donner une mesure plus

précise du niveau de charge et estimer le temps restant avant qu'une recharge soit nécessaire [30]. Bien que cette technique soit utile, elle ne prolonge pas la durée de vie de la batterie. Une technique prometteuse pourrait être les piles à combustible. Une pile à combustible est un dispositif électrochimique qui convertit l'énergie chimique d'un combustible directement en énergie utilisable - électricité et chaleur - sans combustion. Le potentiel énergétique est très élevé, une pile à combustible fonctionnant au méthanol pourrait fournir de l'énergie pendant plus de 20 fois plus longtemps que les batteries traditionnelles au nickel-cadmium dans un emballage de taille comparable [26]. Ils sont théoriquement silencieux et propres comme des batteries normales. Un autre avantage est que les piles à combustible ne nécessitent pas de longues recharges; ils peuvent plutôt être remplis rapidement, simplement en ajoutant plus de carburant. Les piles à combustible étaient autrefois prohibitives. Mais l'ingénierie sophistiquée a récemment entraîné une baisse considérable des coûts. Cependant, la conception d'une pile à combustible miniature pouvant être fabriquée en série à moindre coût est une tâche formidable. Bien que les prototypes initiaux aient été construits, personne n'a encore démontré un appareil compact pouvant être fabriqué en série à un coût inférieur à celui des batteries rechargeables comparables.

Plusieurs chercheurs ont étudié le modèle de consommation d'énergie des ordinateurs portables. Les ordinateurs portables utilisent plusieurs techniques pour réduire cette consommation d'énergie, principalement en les éteignant après une période de non utilisation, ou en abaissant la fréquence d'horloge. Cependant, parce qu'ils ont étudié différentes plates-formes, leurs résultats ne sont pas toujours d'accord, et parfois même contradictoires. Lorch a signalé que l'utilisation d'énergie d'un ordinateur portable typique est dominée par le rétro éclairage de l'écran, le disque et le processeur [28]. Stemm et al. A conclu que l'interface réseau consomme au moins la même quantité d'énergie que le reste du système (c'est-à-dire un PDA Newton) [29]. Si l'ordinateur peut recevoir des messages du réseau même lorsqu'il est éteint, la consommation d'énergie augmente considérablement. Ikeda et al. Observé que la contribution de la CPU et de la mémoire à la consommation d'énergie a été à la hausse ces dernières années [27].

Tableau II.3 : Les potentiels énergétiques de la technologie de batteries actuelles

Batterie	Rechargeable	Wh/kg	Wk/litre
Alkaline MnO2	Non	130	347
Li/MnO2	Non	210	550
Zinc Air	Non	280	1150
Lead acid	Oui	30	80
Nickel-Cadmium NiCd	Oui	40	130
Nickel-metal hybride NiMH	Oui	60	200
Lithium-ion	Oui	60	200
Methanol fuel cell	Oui	6200	4900

II.11 Conclusion :

Enfin, l'ordonnancement processeur doit s'étudier davantage en considérant les autres composants du système : mémoires et périphériques comme ASIC (Application Specific Integrated Circuit)) et DSP (Digital Signal Processor). Par exemple, réduire la fréquence CPU peut augmenter le temps d'utilisation des périphériques et donc leur consommation .Là encore, des compromis sont à trouver entre DVS au niveau du processeur et DPM (mode veille) au niveau des périphériques. Il faut également intégrer le fait que les entrées/sorties avec les périphériques sont généralement d'une durée fixe et que le temps d'exécution de certaines parties du code est donc presque indépendant de la fréquence de fonctionnement du processeur.

CHAPITRE III

➢ *Les résultats de l'analyse de la puissance*

III.1 Introduction :

Les résultats de l'analyse de puissance finale pour l'ensemble de la puce étaient dans la cible de conception de 58W. La puissance de fuite s'est avérée être d'environ 10% de la puissance totale consommée par la puce. Nous notons ici que cette analyse de puissance a été effectuée dans le coin du processus typique et que les résultats obtenus ne s'appliquent pas nécessairement à toutes les puces compte tenu de la variation des caractéristiques dans les copeaux fabriqués. Cependant, en appliquant la méthode de la tension d'alimentation adaptative (ASV) pour ajuster la tension appliquée à chaque puce, la consommation d'énergie moyenne par puce sur l'ensemble d'un système de supercalculateur peut être établie pour atteindre la cible de 58W. Une analyse de puissance détaillée du processeur a été entreprise (avant que le silicium soit disponible) pour fournir une base pour l'amélioration ultérieure l'analyse est basée sur le simulateur parce qu'il n'y a aucun autre moyen d'obtenir une panne de puissance du processeur à un niveau arbitraire. Était PowerMill qui est un outil bien connu utilisé largement pour estimer la consommation d'énergie le processeur de résultats mesurés étaient quelque peu différent des résultats de la simulation, le processeur réel étant plus lent et consommant moins d'énergie que prévu. Malheureusement, aucune donnée sur le cycle de fabrication particulier n'a pu être obtenue, ce qui pourrait expliquer cela. Néanmoins, les résultats de la simulation restent utiles pour la contribution relative à la consommation d'énergie des différentes parties. Comme toutes les parties du processeur sont conçues de la même manière, l'erreur de simulation doit être répartie de façon assez uniforme entre les sous-circuits. La puissance consommée par un processeur dépend des opérations qu'il exécute. Malheureusement, il n'y a pas de référence de facto utilisée pour mesurer la puissance du processeur en termes de performances (par exemple SPEC). De nombreux résultats publiés sont basés sur Dhrystone qui est une référence synthétique qui prétend avoir un mélange dynamique d'instructions similaires aux programmes d'application «typiques». Pour faciliter la comparaison avec d'autres processeurs, les résultats de l'utilisation de Dhrystone seront les plus utilisés ici. D'autres programmes utilisés dans cette analyse étaient DES encryptions / décryptage et un filtre de synthèse à court terme d'une implémentation de GSM. Tous les benchmarks sont écrits dans le langage C et ont été compilés pour la vitesse en utilisant la version 2.5.1 de boîte à outils ARM. Le temps de simulation et la petite taille de la mémoire (8 k octets) dans l'AMULER3i ont été les principaux facteurs limitant dans la sélection de l'indice de référence simulation de niveau de transistor complet a été choisi pour obtenir les résultats les plus précis possibles.

III.2 Le principe de Dhrystone au niveau de la panne de puissance :

Le temps de simulation long résultant limite le nombre d'instructions qui peuvent être simulées dans un délai raisonnable la panne d'alimentation du système de niveau supérieur pour Dhrystone est montré dans (Figure III.1**)** et le noyau dans le grand consommateur responsable d'environ 60% de la puissance Dhrystone et plus pour les autres références (jusqu'à 73%). La RAM de 8 kilo-octets consomme de 20% (filtre) à 30% (Dhrystone). Le reste, qui en moins de 10%, est principalement consommé par le bus système, les périphériques asynchrones n'étaient pas actifs dans les benchmarks. La panne de courant dans le noyau AMULET3 courir Dhrystone est présenté dans (Figure III.1**)** dans cette simulation toutes les performances améliorant les options ont été activées, y compris la branche matérielle de prédiction. Les plus gros consommateurs sont été trouvés être l'exécution et le registre bloc, suivi de l'unité de la prédiction de branche est activée et le décodage bloc.

Le pourcentage utilisé par l'unité de préfecture chute à 13% lorsque la prédiction de branchement est désactivée et que la consommation d'énergie du noyau est inférieure de 6%. Le temps d'exécution augmente seulement un peu (1%), car la méthode de prédiction de branchement utilisée ne fonctionne bien pour Dhrystone. La situation est inversée lors de l'exécution du benchmark du filtre GSM, où la prédiction de branchement fonctionne très bien, supprimant de nombreuses erreurs d'instruction et augmentant les performances de 8%

Le bloc appelé «registres» contient le fichier de registre et les repères de tampon de réorganisation à l'exception du filtre GSM, où l'utilisation intensive du multiplicateur fait de l'unité d'exécution le principal consommateur [37].

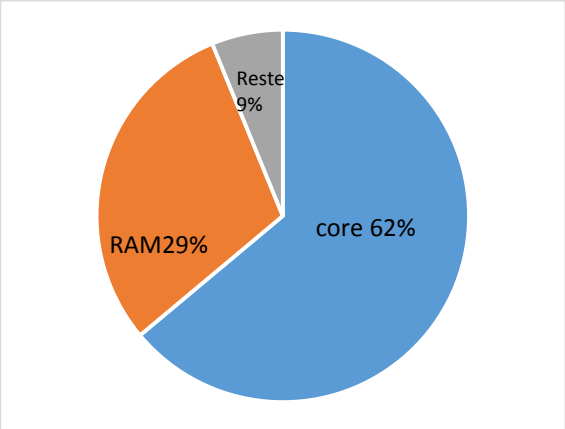


Figure III.1 : Panne de puissance du processeur (Dhrystone)

III.3 Analyse de consommation de puissance :

Les résultats des analyses de puissance initiale et finale pour chaque unité fonctionnelle dans un cœur au cours de la conception utilisant le flux d'analyse de puissance décrit ci-dessus sont présentés à la (Figure III.2**).** La consommation de puissance d'un cœur a été réduite d'environ 6 W à un stade final. En particulier, la puissance consommée par les fichiers de registre et les macros RAM dans chaque élément de circuit a été réduite en utilisant les techniques de réduction de puissance présentées dans la section précédente, et la puissance d'horloge et la puissance utilisées par les sections logiques ont été réduites. Tout au long de la puce en utilisant horloge de synchronisation et d'autres techniques. [37]

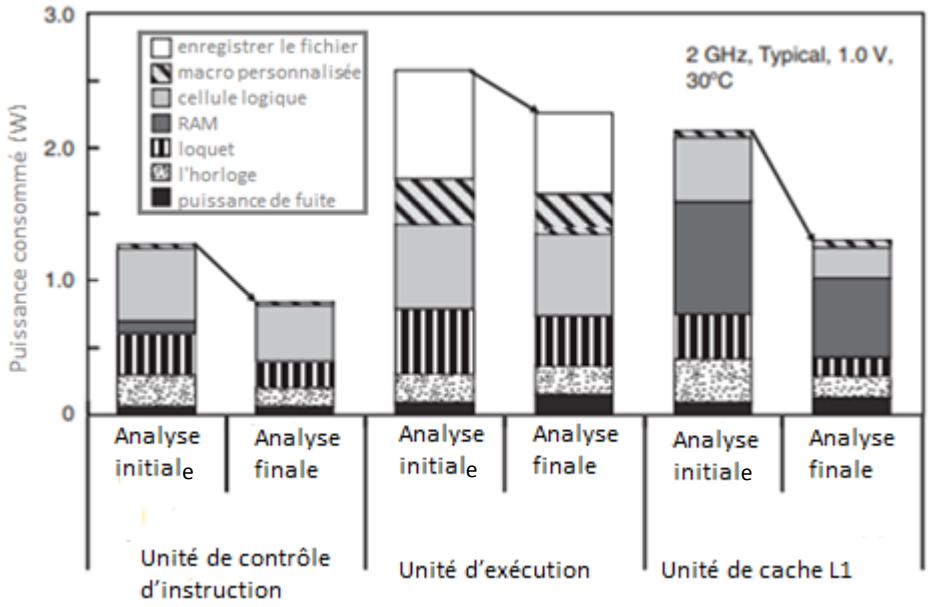


Figure III.2 : Résultats de l'analyse de puissance pour le noyau du processeur

Analyse de la puissance dans la conception de la puce SPARC64 VII perméable en utilisant le taux de fonctionnement moyen en unités de modules de fonction. Cette approche a permis de déterminer la consommation d'énergie pour chaque module fonctionnel, mais pas pour les cellules ou macros individuelles. Pour la puce SPARC64 VIIIfx, nous avons introduit un flux d'analyse de puissance.

Tableau III.1 : SPARC64 VIIIfx chip spécifications

Architecture	SRARC-V9/HPC-ACE SIMD instruction 256 FPRS
	8 cœurs
	32 KB L1$, 32 KB L1D$
	6Mb commun L2$
	Fréquence d'horloge 2GHz
Performance de pointe	performance computationnelle 128Gflops
	débit de la mémoire : 64GB/S
Autre	FSL 45-nm CMOS
	22.7*22.6 mm
	960M transistors
	1271 pins signal

à la porte dans le but d'obtenir une réduction drastique de la puissance de consommation (Figure III.3). Les informations de consommation de puissance obtenues à partir des résultats de cette analyse de puissance en unités de cellules et de macros ont été utilisées dans le processus de conception pour isoler les endroits où les mesures de réduction de la consommation de puissance étaient nécessaires et pour ensuite mettre en œuvre ces mesures et vérifier leurs effets. Les informations de taux de fonctionnement utilisées dans cette analyse de puissance ont été obtenues en unités de cycles utilisant une émulation logique basée sur un émulateur logique. Les informations de fonctionnement elles-mêmes consistaient en taux d'exploitation pour tous les réseaux et broches de la puce. En outre, pour conserver la taille du fichier d'informations d'exploitation généré dans cette analyse et le temps nécessaire pour effectuer la puissance, nous avons sélectionné - depuis toute la période d'émulation logique - plusieurs intervalles d'environ 3000 cycles pour lesquels le taux d'utilisation des unités d'exécution était stable à la valeur maximale et nous avons centré notre analyse sur ces entrevaux pour obtenir des informations opérationnelles détaillées.

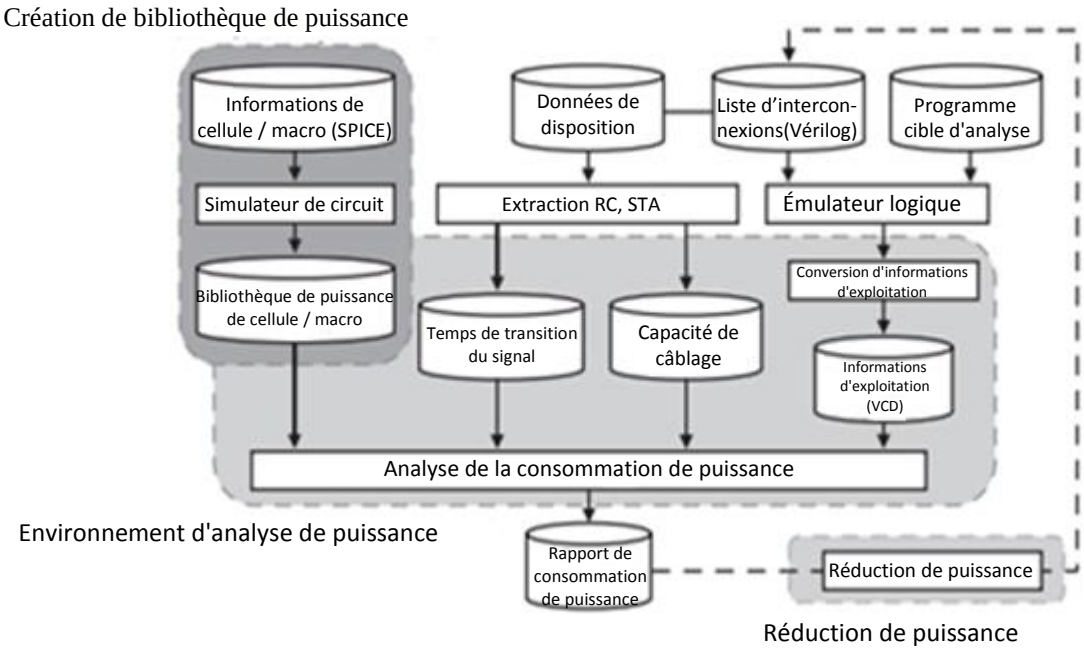


Figure III.3: flux d'analyse de puissance

III.3.1 Analyse de conception de puissance par un flux :

Dans le cadre de ce flux d'analyse de puissance, nous avons créé une bibliothèque de puissance, qui est un fichier qui définit les informations relatives à la consommation de puissance des éléments de circuit individuels, c'est-à-dire, des cellules et des macros. L'analyse de la puissance au niveau de la porte peut calculer la consommation de puissance des cellules et des macros sur la base des informations définies dans la bibliothèque de puissance. Nous créons une bibliothèque de puissance basée sur les résultats de simulation de circuit de cellules standard, de cellules d'E / S et de macros RAM parmi les cellules et les macros pouvant être utilisées pour l'analyse de puissance de ces puces. Cependant, comme cette puce a été développée par conception au niveau de la porte plutôt que par conception de niveau de transfert de registre (RTL), le nombre de macros personnalisées pour augmenter la vitesse et atteindre d'autres objectifs était exceptionnellement élevé, ce qui signifiait qu'il était irréaliste de créer une bibliothèque très puissante pour toutes ces macros. Nous avons donc créé une bibliothèque simple pour les macros personnalisées qui est basée sur des estimations de puissance à partir de grandeurs physiques. En effet, nous avons utilisé la puissance de fuite et la dynamique estimée à partir de la largeur totale du transistor pour chaque type de transistor. Pour la puissance dynamique, nous avons créé une bibliothèque en l'attribuant à des broches macro comme puissance de commutation et, pour les macros de circuits séquentiels, nous avons traité un pourcentage fixe de la largeur totale du transistor comme un loquet et attribué la valeur de cette

macro comme puissance consommée pendant le fonctionnement. En ce qui concerne les macros de registres, nous avons commencé à créer une bibliothèque de puissance comme nous l'avons fait pour les macros personnalisées, mais comme la macro consommation par unité obtenue à la suite de l'analyse de puissance était extrêmement grande par rapport aux macros personnalisées, nous avons décidé de créer une bibliothèque de puissance très précise basée sur les valeurs de puissance obtenues à partir de la simulation de circuit de la même manière que les macros RAM.

Comme cette puce est un produit de conception de niveau porte, il existe de nombreuses cellules avec des structures relativement compliquées ayant des configurations CMOS multi-étages telles qu'un signal multi-entrée ET, un signal multi-entrée OU, un signal multi-entrée ET-OU. Nous considérons qu'une telle cellule doit être divisée en énergie consommée par des circuits internes connectés aux broches d'entrée et consommée par des circuits internes connectés à la broche de sortie. Nous définissons la puissance consommée alors qu'un circuit du côté de l'entrée fonctionne dans les thermiques des broches d'entrée connectées par le circuit et définit la puissance consommée par le circuit côté sortie en termes de broche de sortie connectée à ce circuit et crée une bibliothèque de puissance en conséquence. De cette manière, la puissance ne sera pas sous-estimée même dans le cas d'un fonctionnement de cellule sans transition de sortie. Lors de la conception de cette puce, nous effectuons une analyse de puissance performante à plusieurs reprises sur l'ensemble de la puce.

De plus, en utilisant les résultats après chaque analyse, nous avons vérifié le degré d'atteinte de la cible de puissance pour chaque bloc fonctionnel et révisé les mesures en conséquence. Dans la conception d'un bloc fonctionnel, nous avons progressivement réduit la consommation d'énergie de l'ordre de µW à mW vers la valeur cible établie en faisant des améliorations d'économie d'énergie du niveau de micro architecture au niveau de la cellule / macros. [37]

III.4 Les améliorations de performance du processeur :

Shekhar Borkar et Andrew A. Chien identifient trois domaines principaux qui ont permis à ce jour l'amélioration des microprocesseurs :

L'accélération de la commutation des transistors par la réduction de leur taille : tous les deux ans depuis 40 ans, la taille des transistors a diminué de 30%, ce qui a réduit leur surface de 50%, en d'autres termes doublé la densité du circuit. Cette réduction de la taille du circuit

accroît les performances de 40%. Afin de maintenir un champ électrique constant, la tension est réduite de 30%, ce qui diminue l'énergie consommée de 65%, et la puissance de 50%.

Chaque génération (deux ans) produit donc des processeurs 40% plus rapides, pour un budget énergétique équivalent.

Le perfectionnement de la micro architecture par des techniques telles que :

- Le pipeline, qui consiste à décomposer les instructions en opérations plus élémentaires, ce qui permet de commencer à exécuter une instruction avant que les précédentes ne soient terminées [35].
- L'architecture super-scalaire, qui consiste à multiplier les unités d'exécutions pour les opérations les plus fréquentes, ce qui permet un certain parallélisme.
- L'exécution spéculative et l'exécution « dans le désordre » *(***outof order***)*, qui consistent à exécuter des instructions « à l'avance », alors que l'on n'est pas sûr qu'elles aillent être utiles, mais qui procurent des gains de temps d'exécution significatifs si la prédiction est judicieuse.

L'organisation de la mémoire-cache : le mot cache, curieux retour au français d'un emprunt anglais, suggère ici l'idée de cacher dans un coin (techniquement parlant, dans une zone de mémoire petite mais à accès très rapide) pour l'avoir sous la main une chose que l'on ne veut pas avoir à aller chercher à la cave ou au grenier (i.e., dans la mémoire centrale, vaste mais à accès lent par rapport à la vitesse du processeur), pour gagner du temps. Cette technique s'est développée parce que, si la capacité de la mémoire centrale a évolué parallèlement à la puissance des processeurs, sa vitesse a progressé moins vite. Siau début des années 1990 le temps d'accès à une position de mémoire se mesurait en dizaines de cycles de processeurs, vingt ans plus tard c'est en centaines de cycles : il a fallu trouver des procédés pour éviter que cette discordance croissante ne réduise à néant les bénéfices obtenus pas l'accélération des processeurs. La technique du cache procure des augmentations de performances très spectaculaires, bien qu'aucun modèle général satisfaisant de son fonctionnement n'ait pu être proposé à ce jour. Une variante en est le TLB (Translation Lookaside qui conserve « sous la main » les résultats les plus récents de traductions d'adresses virtuelles en adresses réelles, ce qui permet avec un très faible volume de données d'obtenir des accélérations spectaculaires [36].

III.5 Gisements d'amélioration en voie d'épuisement :

Les voies classiques d'amélioration que nous venons de survoler continueront à être exploitées, mais depuis quelques années elles sont de moins en moins fructueuses, ne serait que parce qu'elles ont atteint certaines limites. Ainsi, la réduction de la taille des transistors approche de limites physiques : la géométrie des processeurs annoncés récemment (2012) repose sur des motifs de 22nanomètres, c'est-à-dire que l'épaisseur du diélectrique des transistors est de quelques dizaines d'atomes. On ne pourra pas descendre beaucoup plus bas. De toute façon, après plus de trente ans de progrès exponentiels, la courbe des gains de performances en fonction de la taille des transistors a commencé à s'infléchir depuis cinq ou six ans. Par exemple, la tension d'alimentation diminue avec la taille des circuits, mais la diminution de la tension de seuil de commutation du transistor augmente le taux de fuite (le courant qui « passe » quand le transistor est dans l'état « non passant »), ce qui oblige à maintenir un seuil de tension de commutation moins bas, et ainsi à limiter le gain de performance espéré. Mais le principal obstacle auquel les concepteurs de circuits se heurtent aujourd'hui est la consommation électrique : alors que pendant les trente premières années du microprocesseur on ne s'en souciait guère.

C'est devenu une préoccupation essentielle, et pas uniquement pour des raisons écologiques. Que l'on songe au fait qu'une compagnie comme Google soit amenée à investir des centaines de millions de dollars dans l'énergie photovoltaïque et à installer ses fermes de serveurs au pied des centrales pour réduire la facture énergétique. Avec l'augmentation de la densité des composants et la réduction de la taille des machines, la dissipation calorique devient un problème crucial pour les data centres, nouveaux noms des centres de calcul. C'est en partie, mais pas seulement, pour des raisons de consommation électrique que le nombre d'étages des pipelines a commencé à diminuer. Le processeur Intel Pentium 4, par exemple, introduit en 2000, avait une géométrie de 65 nm et un pipeline de 31 étages, ce qui lui permettait d'afficher une fréquence d'horloge de 3,8 GHz. Cette fréquence élevée était en partie un élément de marketing, destiné à impressionner le public, parce que le gain marginal procuré par le dernier étage du pipeline est faible. Aujourd'hui les fréquences n'augmentent plus guère et on est revenu à des pipelines plus raisonnables, 16 étages pour l'architecture Nehalem à la base des processeurs Core i7 sortis en 2008.

III.6 Nouvelles orientations du progrès :

Les processeurs les plus récents ont plus de deux milliards de transistors sur deux ou trois centimètres carrés : c'est plus qu'il n'en faut pour implanter toute la logique nécessaire, alors on fait des processeurs multi-cœurs, c'est-à-dire en fait des multiprocesseurs (chaque cœur est un processeur simple), avec huit cœurs ou plus sur un même chip. Même ainsi, il reste plein de transistors disponibles : on les utilise pour installer de la mémoire, dont le temps d'accès sera ainsi excellent. On peut aussi embarquer à bord des fonctions qui étaient auparavant affectées à des composants séparés : décodage vidéo, cryptographie, réseau...Mais la recherche s'oriente vers une utilisation plus subtile de cet énorme stock de transistors disponibles. Puisque qu'ils sont en abondance, et qu'en outre les utiliser tous ensemble à leur fréquence maximum entraînerait une consommation électrique et une dissipation thermique intolérables (l'extrapol at-ion des tendances actuelles donnerait 500 W par centimètre carré en 2018), l'idée est, plutôt que de multiplier les cœurs généralistes identiques les uns aux autres, d'en mettre moins, qui seront associés à de la mémoire, moins consommatrice d'énergie, et aussi d'implanter des cœurs spécialisés pour certains traitements, qui ne serviront peut-être pas souvent, mais qui donneront de bons résultats quand on en aura besoin.

D'autres possibilités sont explorées : mieux organiser les accès à la mémoire de façon à placer les données le plus près possible des circuits qui doivent y accéder, hiérarchiser les lignes de communication(bus) entre les groupes de cœurs et optimiser le placement des tâches sur les cœurs en fonction des accès aux données, voir utiliser une partie du budget de transistors pour des circuits logiques programmables(Field-programmable Gate Array, FPGA), c'est à dire du matériel modifiable à la volée par l'utilisateur, ce qui existe déjà, mais avec des performances inférieures à celles des microprocesseurs actuels.

III.7 Décomposition en facteurs de la loi de Moore :

La plupart des processeurs modernes peuvent adapter leur fréquence d'horloge et leur tension d'alimentation à la tâche qu'ils effectuent à un instant donné, ce qui complique l'étude de l'évolution de ces variables au fil des améliorations de la technologie.

Afin de pouvoir comparer raisonnablement les performances de processeurs de conception et de réalisation technique très différentes, à cause principalement des progrès des procédés de

fabrication, nos auteurs ont dû mettre au point une méthodologie de normalisation de la technologie. Pour ce faire ils ont cherché à estimer le facteur d'échelle procuré par la technologie à la performance. Il leur fallait définir une métrique pour la performance : ils se sont appuyés surtout sur les résultats de la suite de benchmark SPEC 2006, et pour les processeurs trop anciens, sur les versions antérieures de SPEC (1989, 1992, 1995 et 2000). Pour établir des abaques de passage d'une métrique à une autre ils se sont appuyés sur l'existence de processeurs pour lesquels étaient disponibles des résultats dans deux échelles successives. Pour estimer la performance que procurerait un processeur ancien s'il était réalisé selon une technologie plus moderne, le facteur fréquence d'horloge est certes déterminant, mais il doit être pondéré par le temps d'accès à la mémoire, qui a progressé beaucoup moins rapidement. Pour calculer une puissance putative, les auteurs supposent que les constructeurs augmentent la taille de la mémoire cache "on-chip" afin de maintenir constant le taux de défauts de cache, c'est-à-dire la proportion des tentatives d'accès aux données qui ne sont pas satisfaites par le cache, et qui nécessitent un accès réel à la mémoire centrale, avec la mise en attente qui en résulte pour le processus concerné. Ils en déduisent la loi empirique selon laquelle la taille du cache doit croître comme le carré de la fréquence d'horloge. Pour imputer une fréquence d'horloge putative à un processeur ancien s'il était réalisé selon une technologie plus moderne, il faut aussi connaître l'évolution des délais induits par les portes logiques et parles fils de liaison. Ces données sont heureusement sensiblement les mêmes pour tous les types de portes, et fréquemment disponibles, ce qui a permis aux auteurs d'établir une formule d'extrapolation.

III 7.1 Le concept du développement du processeur :

Le rythme d'introduction de nouveaux procédés de fabrication, permettant une nouvelle géométrie des circuits, est de deux à trois ans, avec une certaine accélération dans la dernière décennie. Chaque génération divise la taille des transistors par $\sqrt{2}$. Au cours des 25 ans écoulés depuis l'Intel 80386, la taille des transistors ça a été réduite selon un facteur 4 000 ; or le nombre de transistors par processeur a été multiplié par 16000 : l'explication est l'augmentation de la surface des processeurs, 103 mm² pour le 80386, 296 mm² pour l'Intel Core i7 par exemple.

Si le seul facteur d'augmentation de la fréquence était la taille des transistors, les processeurs actuels tourneraient à 500 MHz, au lieu de 3 GHz et plus : un autre facteur est l'augmentation du nombre d'étages des pipelines, c'est-à-dire que chaque opération est décomposée en un plus grand nombre de micro-opérations. Le 80386 avait un pipeline à deux

étages (une étape "Fetch", une étape "Exécute"), 31 étages pour le Pentium IV, et un retour vers des valeurs plus raisonnables avec 16 étages pour le Core i7.Si la puissance électrique avait crû comme la fréquence, le facteur aurait été plus important. Entre le 80386 (1986) et le Pentium 4 (2000), la taille des transistors a été divisée par 16, la tension d'alimentation divisée par 4, la fréquence d'horloge multipliée par 200. On aurait pu s'attendre à une multiplication de la puissance électrique nécessaire par $16\times200/4^2=200$. Or le facteur observé est seulement 32. Cette économie est essentiellement procurée par des optimisations de l'architecture, notamment dans la propagation des signaux d'horloge.

III.8 Mesure de la consommation de puissance:

Même avec un matériel homogène, la variabilité inter-nœuds est toujours présente. Nous mesurons la consommation de puissance chaque seconde par nœud à l'aide du protocole IPMI (Intelligent Platform Management Interface). Les différents serveurs lame utilisés comme systèmes de test présentent des consommations de puissance de 182W-196W (une différence de plus de 5%), même lorsqu'ils sont complètement inactifs. Lorsque le RAM de différents fournisseurs était utilisé, les différences étaient encore plus grandes. Nos tests indiquent que la variation de la consommation d'énergie est étroitement liée aux baies de périphériques physiques de l'enceinte du serveur. Les problèmes thermiques sont peu susceptibles d'être le coupable, car les vitesses des ventilateurs dans l'ensemble de l'enceinte ne varient que légèrement.

Pour compenser cette variabilité, toutes les mesures de consommation de puissance sont moyennées sur un minimum de 15 essais. Ces analyses sont à leur tour réparties sur au moins trois nœuds différents. En utilisant cette approche, nous évitons le scénario où différents algorithmes et configurations se voient attribuer un avantage ou un empêchement injustifié simplement en raison de l'attribution d'un nœud. Les moyennes constituent également une bonne mesure de la performance et de la consommation de puissance lorsque l'on travaille avec un grand nombre de serveurs, comme on peut s'y attendre dans un environnement à courant continu.

Conclusion :

L'amélioration continue dans tous les compartiments**,** depuis 2005 l'évolution est commandée par la recherche d'économie d'énergie, notamment par l'implantation sur un même chip de plusieurs cœurs, c'est-à-dire des processeurs élémentaires, ce qui autorise du traitement parallèle sur un seul chip. Mais néanmoins la performance unitaire des cœurs continue à augmenter, même indépendamment des apports de l'accroissement de la fréquence et de la taille du cache. Ces gains proviennent de perfectionnements architecturaux tels que l'implantation sur le chip des contrôleurs d'accès à la mémoire et la multiplication des unités d'exécution (architecture super-scalaire).Il est toutefois difficile d'isoler ces progrès dus au matériel de ceux dus aux progrès des compilateurs, dont dépendent les résultats des benchmarks SPEC. La base de données CPU permet désormais de quantifier les progrès incroyables réalisés en quarante ans par l'industrie des microprocesseurs, et offre les moyens d'en analyser les évolutions et les tendances par des méthodes scientifiques.

CHAPITRE IV

➢ *Comparaison avec les mesures de la puissance*

Introduction :

Aujourd'hui, les applications portables se démocratisent, elles requièrent de plus en plus de puissance de traitement ce qui augmente la consommation d'énergie de tels systèmes. Pour éviter d'avoir à sur dimensionner les batteries, il faut donc optimiser la consommation de ces applications. Ces optimisations peuvent se faire à tous les niveaux d'abstraction mais il existe un point commun entre elles : le besoin d'avoir une métrique de consommation rapide et fiable sans devoir faire de mesures physiques.

Les développeurs utilisent de plus en plus le langage C pour décrire leurs applications, il est donc intéressant d'estimer directement la consommation de l'application à ce niveau. Ceci évite de passer par une phase de compilation donc augmente l'interactivité entre l'estimation de consommation et le développement de l'application. Des outils existent déjà pour les cœurs de processeur ou pour les ASICs (Wattch, Jouletrack, outils constructeurs etc.). En revanche, très peu d'études sont réalisées sur les processeurs commerciaux comme les DSP. Le problème avec ces cibles est le manque d'informations disponibles sur l'architecture et son fonctionnement.

Avec cette méthode, il faut mesurer la consommation de tout le jeu d'instructions ainsi que de toutes les inter-instructions, ce qui représente un nombre prohibitif de mesures pour des architectures complexes récentes.

IV.1 Comparaison avec les mesures de puissance :

Modèle d'énergie analytique dans le simulateur de Wattch [38]

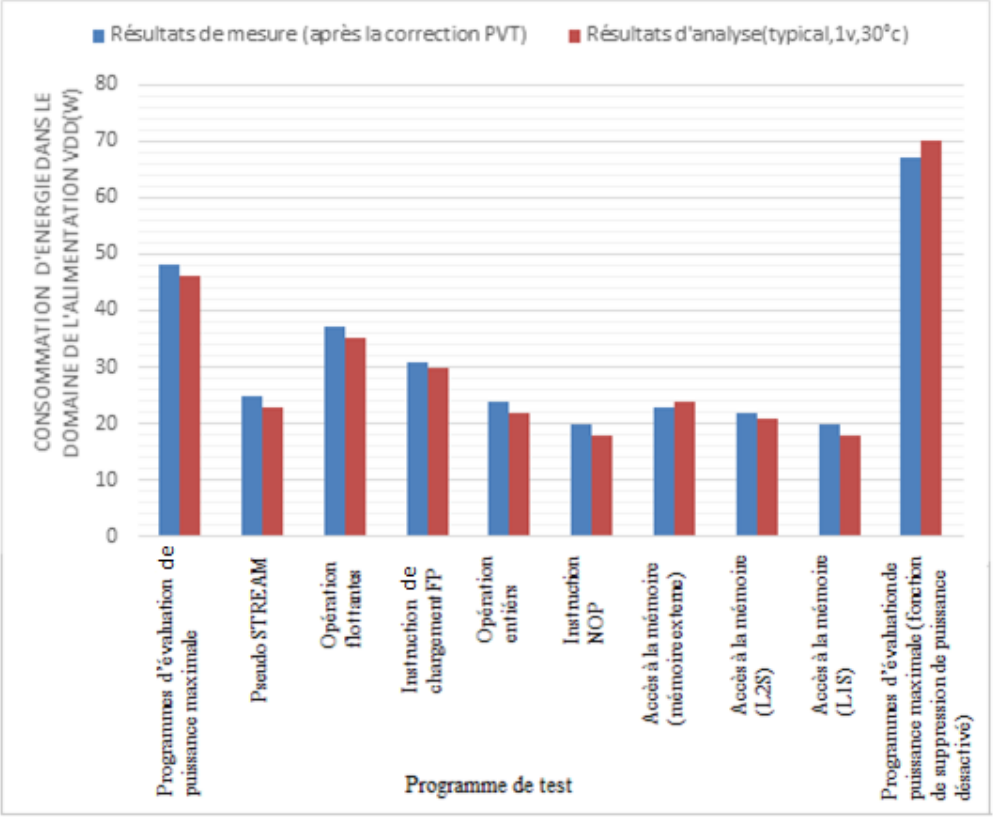


Figure IV.1 : Comparaison entre les résultats de mesure de puissance et d'analyse de puissance[38]

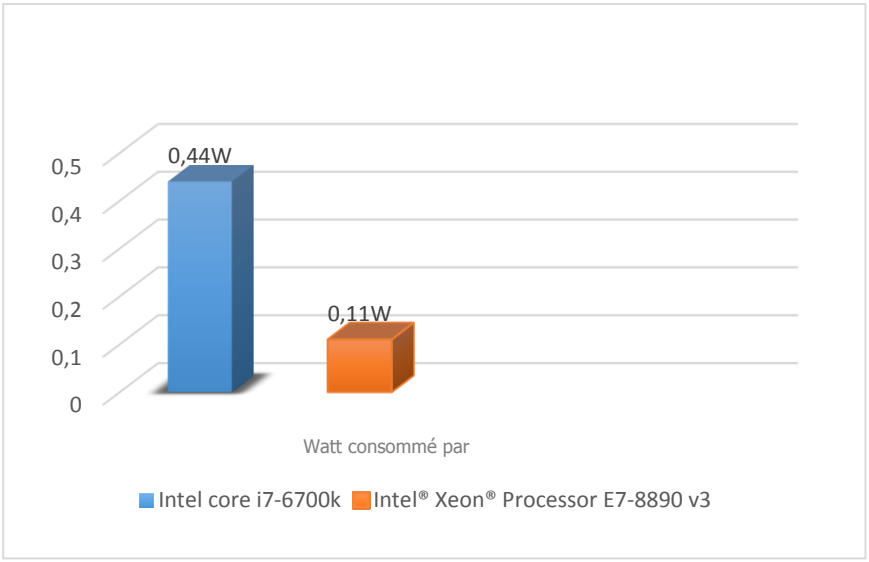


Figure IV.2 : Comparaison de la consommation électrique par GigaFlop[38]

IV.2 Définition de FLOP :

FLOP (Floating-point operations per second).Opérations en virgule flottante par seconde. Mesure de la vitesse d'un processeur ou d'une des unités de calcul arithmétique d'un processeur. Une opération en virgule flottante manipule des nombres décimaux à virgule, codés sur un certain nombre de bits.

Cette unité est mieux représentative de la puissance de calcul d'un processeur que sa fréquence (exprimée en hertz, Hz) ou encore que le nombre d'instructions par seconde qu'il est capable d'exécuter (Mips, millions of instructions per second).Un préfixe indique toujours une quantité : mégaflops (10^6 flops), gigaflops (10^9), téraflops (10^{12})... Les puissances des processeurs actuels se mesurent en général en gigaflops.

IV.3 Comparaison avec un système synchrone :

Dans un processeur synchrone, une grande partie de la puissance est attribuée à l'horloge. Tiwari et al [38] indiquent qu'environ 40% de la puissance du processeur est dépensée sur l'horloge, y compris le générateur, pilotes, arbre de distribution et de chargement. StrongARM [40], avec une architecture similaire à AMULET3, est signalé à utiliser 26% de sa puissance dans l'horloge, y compris le PLL. Autres processeurs état des résultats similaires.

Il serait intéressant d'estimer l'équivalent de cette puissance dans AMULET3.Comme il n'y a pas de moyen simple de déterminer quel circuit doit être considéré comme équivalent à l'horloge, le choix est quelque peu arbitraire. L'équivalent placard est l'ensemble des contrôleurs de verrouillage du pipeline verrous entre les sous-modules du cœur, ceux-ci comprennent les pilotes pour les grandes charges de validation de verrouillage. En outre, certains pilotes de signaux de pré charge contrôlés par des signaux de prise de contact ont également été inclus dans ce groupe. La contribution de tous ces circuits à la consommation de puissance s'est révélée être de 10,5% du cœur lors de l'exécution de Dhrystone 2.1.La comparaison avec la proportion de la puissance prise par une horloge montre que les techniques asynchrones peuvent réduire considérablement la consommation de puissance. Naturellement, les avantages du style de conception asynchrone ont un coût. Comme les différentes étapes du pipeline ne sont pas synchronisées, les informations d'état entre les étapes du pipeline sont difficiles à échanger. Cela conduit à la duplication des informations dans plusieurs endroits du pipeline. Dans

AMULET3, chaque étape du pipeline conserve l'adresse de l'instruction en cours de traitement en raison de la difficulté d'accès à un PC central. Ce n'est pas une puissance significative entendue, car seulement quelques bits du PC commutent à chaque fois. En plus de dupliquer l'information, le contrôle du grain fin des circuits conduit à l'existence de plus d'informations d'état / séquençage par rapport à un processeur synchrone. Augmentation de la puissance de contrôle pour le processeur, proportionnellement environ 40% de la puissance de base du processeur dans AMULET3.Ceci est assez élevé par rapport aux autres résultats publiés, même si cela inclut la plupart de la "horloge" puissance indiquée ci-dessus. Il faut noter cependant que les circuits de contrôle sont implémentés en utilisant des cellules standard, placés automatiquement et acheminés par des outils de CAO, tandis que les datapathes sont entièrement personnalisés. Ainsi, les capacités de fil tendent à être plus élevées et il y a moins de contrôle sur la force d'entraînement des portes. Comme les processeurs synchrones de faible puissance sont leurs unités de contrôle ont tendance à être aussi compliqué que leurs homologues asynchrones. De plus, l'architecture ARM est assez complexe pour une machine RISC qui rend la logique de contrôle plus difficile et énergivore, ARM7 [41], qui implémente la version précédente de l'architecture ARM, commence à consommer 40% de sa puissance dans la partie contrôle, bien que la définition des circuits de contrôle puisse être différente. [42]

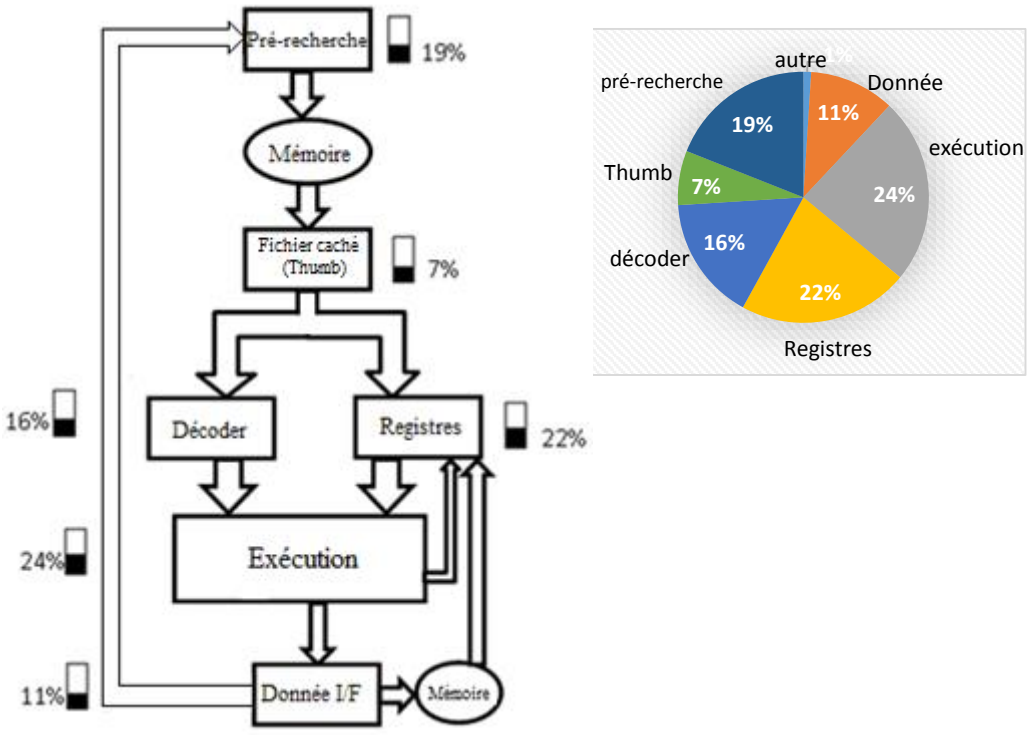


Figure IV.3 : AMULET3 consommation de puissance de base [42]

IV.4 Comparaison avec ARM9 :

ARM9TDMI est le placard d'implémentation synchrone d'AMULET3, les deux exécutent le même jeu d'instructions et ont été implémentés sur la même technologie, occupant la même zone de silicium (environ 4mm^2) .Sur ce processus de 0.35µm ARM9 fonctionne jusqu'à 120 MHz avec une performance de 1.1 MIPS / MHz et consommant 1,8 mW / MHz. Cela donne une énergie par mesure d'instruction de 610MIPS / W. Malheureusement, une panne de courant n'a pas été donnée.

Les résultats mesurés par AMULET3i montrent une consommation de puissance de 221mW à la performance de 85 Dhrystone MIPS. Selon les résultats de simulation, le cœur consomme 62% de la puissance (137mW). Ceci donne une énergie par instruction de 620MIPS / W, effectivement le même que celui de ARM9.

La performance est plus lente que prévu d'après la simulation, qui a prédit une vitesse d'environ 100MIPS pour le système (on ne sait pas à quel point d'exécution de silicium était «typique».) De plus, cela était limité par le système de mémoire (non plastifié). Le cœur fonctionne à lui seul à environ 130 MIPS en simulation. Ainsi, il est sûr de supposer une vitesse de plus de 100 MIPS pour le processeur sur silicium, sans les limitations de la mémoire [42]

IV.5 Performance de comparaison des différents produits :

Tableau IV.1 : performance de comparaison (mA /MHz) avec système d'horloge à 80MHz

Configuration	Produits				Commentaire
	STM32L 47X/48X	STM32L 43X/44X	STM32L 45X/46X	STM32L 49X/4AX	
FLASH ART désactivé	0.117	0.100	0.102	0.114	---
FLASH ART activé	0.136	0.121	0.118	0.133	Cache activé, pré-recherche de tampon désactivé
SRAM1	0.130	0.097	0.097	0.121	Code et données dans SRAM1

Les performances des différents produits sont comparées dans le tableau IV.1.

L'ART Accelerator ™ permet d'atteindre presque les mêmes performances, aussi bien en calcul (CoreMark® par MHz) qu'en consommation de courant (CoreMark® par mA), comme si le même programme était exécuté depuis la SRAM interne. [45]

Tableau IV.2: performance de comparaison (mA /MHz) pour les différents modes d'exécutions

Mode d'exécution	Configuration	Produits			
		STM32L 47X/48X	STM32L 43X/44X	STM32L 45X/46X	STM32L 49X/4AX
Range1 (80MHz)	FLASH ART désactivé	0.117	0.100	0.100	0.117
	FLASH ART Activé	0.136	0.121	0.118	0.133
Range2 (26MHz)	FLASH ART désactivé	0.111	0.094	0.096	0.114
	FLASH ART Activé	0.118	0.103	0.102	0.110

Les performances des différents produits sont comparées dans le tableau IV.2.

La sélection de la plage 2, si possible, améliore l'efficacité (CoreMark® par mA) de près de 15%. [45]

IV.6 Comparaison des pannes de courant :

Wattch [43] et SimplePower [44] sont deux autres outils de surveillance de puissance au niveau du processeur basés sur SimpleScalar [39], qui est le simulateur de microarchitecture le plus populaire. Dans Wattch, les modèles d'énergie dépendent des capacités internes des circuits qui composent chaque unité du processeur. Chaque unité modélisée appartient à l'une des quatre catégories suivantes: les structures de matrice, les mémoires, la logique combinatoire et les fils, et le réseau d'horloge. Un modèle de puissance différent est utilisé pour chaque catégorie et intégré dans le simulateur SimpleScalar pour fournir une variété de mesures telles que la puissance, la performance, l'énergie et le produit à retard d'énergie. Le (**tableau IV.3)** montre la dépense énergétique de divers composants à partir des mesures ainsi que du simulateur de Wattch.

Tableau IV.3 : Comparaison des pannes de courant entre les mesures (Alpha 21264) et modèle d'énergie analytique dans le simulateur de Wattch

Structure Matérielle	Mesure (Alpha 21264)	Modèle analytique (Wattch)
Caches	16.1%	15.3%
Problème de logique d'émission	19.3%	20.6%
Mémoire	8.6%	11.7%
Unité de gestion mémoire	10.8%	11.0%
Unité d'exécution à virgule flottante	10.8%	11.0%
Réseau de pointage	34.4%	30.4%

SimplePower, d'autre part, est basé sur un modèle d'énergie sensible à la transition, où chaque unité fonctionnelle modélisée a sa propre capacité de commutation pour chaque transition d'entrée possible [44]. Ceci est ensuite utilisé pour calculer la puissance consommée dans une unité fonctionnelle particulière basée sur la transition d'entrée lors de l'exécution d'une instruction donnée. SimplePower est utilisé pour évaluer l'impact d'une modification architecturale ainsi que l'effet d'une technique d'optimisation de haut niveau. Puissance du système. Des exemples d'utilisation de SimplePower incluent une technique de pipeline sélectif pour réduire la capacité de commutation de données, la transformation de boucle et de données pour réduire la puissance du système de mémoire et le ré-étiquetage pour économiser l'énergie sur les bus de données [44].

IV.7 Comparaison de puissance entre les différents processeurs :

Le tableau ci-dessus montre une comparaison de consommation de puissance entre différents types de processeur (Tableau IV.4):

Tableau IV.4: Comparaison de consommation de puissance du processeur de modèle Intel

Intel i5 CPU			Intel i7 CPU			Intel i9 CPU		
Modèle	Cœurs	Puissance consommée (Watts)	Modèle	Cœurs	Puissance consommée (Watts)	Modèle	Cœurs	Puissance consommée (Watts)
i5-6600k	4	91	i7-4790k	4	88	i9-7200x	10	140
i5-3570k	4	88	i7-6700k	4	91	i9-7920x	12	140
i5-7400k	4	91	i7-7820k	8	140	i9-7980xE	18	165
i5-7400k	4	65	i7-8700	6	65	i9-7960x	16	165
i5-6500k	4	65	i7-8700k	6	95	i9-7940x	14	165

La consommation de puissance des processeurs Intel i5 varie de 65 watts à 95 watts pour les modèles de bureau dans notre base de données.

La puissance du processeur Intel i7 va de 35 watts à 140 watts pour les modèles de bureau dans notre base de données.

La consommation de puissance (en watts dans ce cas) des processeurs Intel i9 varie de 140 à 165 watts, selon le modèle. Sans surprise, les modèles 14, 16 et 18 cœurs consommaient plus d'énergie que ceux avec moins de cœurs, et tous les modèles i9 consommaient plus de puissance que tous les modèles i7 de notre base de données (à l'exception du i7-7820X).

IV.8 Comparaison de performance de puissance/d'énergie d'Intel architectures :

Cette section couvre brièvement deux principales microarchitectures de processeurs publiées par Intel: Sandy Bridge et Haswell. La microarchitecture définit l'organisation des composants, tels que les cœurs, les unités d'exécution ou les caches, à l'intérieur du processeur. Depuis 2007, Intel a adopté un calendrier de diffusion "tick-tock" où une "coche" rétrécit les transistors et un "tock" est la sortie d'une nouvelle microarchitecture.

Sandy Bridge et Haswell sont des "tocks". Nous couvrons également les variantes -EP de ces architectures, principalement destinées aux serveurs.

Tableau IV.5: Comparaison des architectures Intel et des familles de modèles

Nom de famille	Réalisation	Nœud	Cœurs	Cache L3	TDP
Sandy Bridge	2011	32nm	1-8	1-15MB	17-150W
Sandy Bridge-EP	2011	32 nm	2-8	5-20 MB	6-150W
Haswell	2013	22 nm	2-8	2-20MB	11.5-140W
Haswell-EP	2014	22 nm	4-18	10-45 MB	52-165W

Le tableau IV.5 présente une comparaison de quatre familles de modèles par Intel. Nous utilisons certains paramètres clés pour comparer les familles telles que la date de publication, la taille du nœud, le nombre de noyaux, la taille du cache L3 et la puissance de calcul thermique (TDP). En général, nous avons un nombre de cœurs plus élevé, un cache L3 plus grand et un TDP plus élevé. Les modèles -EP comportent également des contrôleurs de mémoire à quatre canaux et manquent de graphiques intégrés.

L'architecture de Sandy Bridge était une version majeure avec des améliorations significatives en termes de performances et de consommation d'énergie. Les nouvelles instructions AVX (Advanced Vector Extensions) améliorent considérablement les performances des applications utilisant des instructions SSE (Streaming SIMD Extensions) auparavant.

En outre, le GPU et le contrôleur de mémoire ont été intégrés sur la même puce de silicium dans Sandy Bridge, ce qui réduit encore plus la latence de la mémoire.
La fonctionnalité « Turbo Boost » a été améliorée par rapport à la précédente microarchitecture Nehalem. Sandy Bridge a également présenté RAPL et le cache de micro-opération. Ces fonctionnalités sont décrites dans les sections 2.6 et 2.7.

Haswell est le successeur de Sandy Bridge. Il présente le jeu d'instructions AVX 2 qui permet des opérations sur des entiers de 256 bits. L'architecture dispose de deux ports d'exécution supplémentaires par rapport à Sandy Bridge, ce qui porte le nombre total de ports

d'exécution à huit. Le GPU intégré a été grandement amélioré dans Haswell. Haswell dispose d'un régulateur de tension entièrement intégré, poursuivant la tendance consistant à intégrer davantage de composants sur la même matrice. L'architecture prend également en charge de nouveaux états de veille à faible consommation appelés C6 et C7. [45]

CONCLUSION GÉNÉRALE

Conclusion Générale :

L'objectif de ce travail a été atteint d'après des recherches et des aides précieuses de notre encadreur et ses expériences qu'il a apportées. Néanmoins quelques difficultés ont été rencontrées pendant la réalisation de notre projet, nous avons eu des difficultés de différentes natures pédagogiques, techniques, organisations et autres :

- L'obtention de certaines données.
- Difficulté organisationnelle entre les membres du groupe

En effet, nous espérons que ce mémoire est arrivé à répondre à la question générale qui est posée au début où passe l'énergie engloutie par nos processeurs ? et au mot « diminuer l'énergie » Et beaucoup de notions de base du système embarqué.

Enfin, pour ce qui reste à faire pour les travaux futurs est d'abord de mettre ce document électronique à la disponibilité des professeurs dans l'optique de le tester et de faire les mises à jour, ensuite d'élaborer d'autres guides électroniques vers le second cycle d'étude supérieure.

RÉFÉRENCE BIBLIOGRAPHIQUE

Référence bibliographique

[1] Https://www.tomsharware.fr/articles/k-computer,1-40225.html.

[2] Https://patents.google.com/petent/EP0904547A1/fr.

[3] Guy Grave, « gestion de la consommation électrique du processeur », 28 décembre 2013.

[4] : PARAIN F., BANÂTRE M., CABILIC G., HIGUERA T., ISSARNY V., LESOT J.-P., Techniques de Réduction de la Consommation dans les Systèmes Embarqués Temps-Réel, Rapport n˚3932, IRISA, mai 2000.

[5]: SU C.-L., DESPAIN A., « Cache Designs for Energy Efficiency », 28th Hawaii International Conference on System Science, p. 306–315, 1995.

[6]: ABNOUS A., SENO K., ICHIKAWA Y., WAN M., RABAEY J., « Evaluation of a Low-Power Reconfigurable DSP Architecture », IPPS/SPDP Workshops, p. 55–60, 1998.

[7] : DAVID R., SENTIEYS O., « Architectures reconfigurables : opportunités pour la faible consommation », 4ième journée d'études Faible Tension Faible Consommation (FTFC'03), p. 31-40, mai 2003.

[8] : REMOND Y., SIRIANNI A., SICARD G., RENAUDIN M., « Estimation et Optimisation de la Consommation d'Energie des Circuits Asynchrones », 4ième journée d'études Faible Tension Faible Consommation (FTFC'03), p. 59-64, mai 2003.

[9]: TIWARI V., MALIK S., WOLFE A., « Power Analysis of Embedded Software : a First Step towards Software Power Minimization », IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 2, n˚4, p. 437–445, 1994.

[10]: J. Rabaey and M. Pedram, "Low Power Design Methodologies", Kluwer Academics, 1996.

[11]: J. Rabaey, "Digital Integrated Circuits, a design perspective", Prentice Hall, 1996.

[12] : M. Renaudin, "Asynchronisme et Adéquation Algorithme Architecture", Journées AAA, janvier 1996.

[13]: E. Macii, "High Level Design and Optimization for Low Power", NATO Advance Study: Low Power in Deep Submicron Electronics, Aug 1996.

[14] G. Qu, N. Kawabe, K. Usami, and M. Potkonjak, "Function-level power estimation methodology for microprocessors," presented at the Proceedings of the 37th Annual Design Automation Conference, Los Angeles, California, USA, 2000.

[15] Rabiner W., Chandrakasan A.: “Network-Driven Motion Estimation for Portable Video Terminals”, Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP ’97), April 1997, Vol. 4, pp. 2865-2868.

[16] (21/05/2013). Texas Instruments. Available: http://www.ti.com/

[17] S. K. Rethinagiri, R. Ben Atitallah, S. Niar, E. Senn, and J. Dekeyser, "Fast and accurate hybrid power estimation methodology for embedded systems," in Design and Architectures for Signal and Image Processing (DASIP), 2011 Conference on, 2011, pp. 1-7.

[18] J. Laurent, N. Julien, E. Senn, and E. Martin, "Functional level power analysis: an efficient approach for modeling the power consumption of complex processors," in Design, Automation and Test in Europe Conference and Exhibition, 2004. Proceedings, 2004, pp. 666-667 Vol.1

[19] : A. Chandrakasan, M. Potkonjak, R. Mehra, J. Rabaey and R. Brodersen, “Optimizing Power Using Transformations”, IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, vol. 14, no. 1, pp. 12-31, 1995.

[20]: A. Abnous, and J. Rabaey, “Ultra Low-Power Domain Specific Multimédia Proceessors”, IEEE Workshop on VLSI Signal Processing, IEEE Press, pp. 461-470, Oct 1997.

[21]: T. Callaway, and E. Swartzlander, “Optimizing Arithmetic Elements for Signal Processing”, IEEE Workshop on VLSI SP, IEEE Press, pp. 91-100, Oct 1992.

[22] : J. Dedou and O. Sentieys, “Synthèse d’Architectures Asynchrones en Traitement Numériques du Signal”, Colloque CAO de circuits intégrés et systèmes, Grenoble, Janvier 1997.

[23]: RABAEY J., PEDRAM M., Eds., Low Power Design Methodologies, Kluwer Academic Publishers, 1996.

[24]: GRUIAN F., Energy-Centric Scheduling for Real-Time Systems, PhD thesis, Lund Institute of Technology, 2002.

[25]: LORCH J., SMITH A., « Improving Dynamic Voltage Scaling Algorithms with PACE », ACM SIGMETRICS 2001 Conference, p. 50–61, 2001.

[26] Dyer C.K.: "Replacing the battery in portable electronics", Scientific American, pp. 70-75, July 1999.

[27] Ikeda T.: “ThinkPad Low-Power Evolution”, IEEE Symposium on Low Power Electronics, October 1994.

[28] Lee M.T.-C et al. V. Tiwari et al.: “Power analysis and low-power scheduling techniques for embedded DSP software”, International Symposium on System Synthesis, pp. 110-115, Sept. 1995.

[29] Stemm, M, et al.: “Reducing power consumption of network interfaces in hand-held devices”, Proceedings mobile multimedia computing MoMuc-3, Princeton, Sept 1996.

[30] Truman T.E., Pering T., Doering R., Brodersen R.W.: The InfoPad multimedia terminal: a portable device for wireless information access”, IEEE transactions on computers, Vol. 47, No. 10, pp. 1073-1087, October 1998.

[31] Web.mit.edu/rhel-doc/4/RH-DOCS/rhel-isa-fr-7/S1-bandwidth-processing.html.

[32] https://www.macg.co/2014/09/un.corrollaire-à-la-loi-de-moore-6182.

[33] Kepeklian.com/la-loi-de-glider-de-passement-de-bande.

[34] Real time digital signal processor based on the TMS320C600 by Nasser Ketlannavaz “Book4you”

[35]Https://Lawrentblock.net/Myspips/les-microprocesseur-leur-histoire-et-leur-avenir?langue fr///Site WWW de Laurent Block.

[36] Y.Kawabe et al..power reduction techniques using in SPARC64 VIIIfx processor for fujitsu’s next generation supercomputer.

[37] Olivier Sentieys “Réduction de consommation d’écriture en électrique embarquée, univ Rennes ;

Avril 1997.

[38] V.Tiwari, et al: Reducing power in high-performance Microprocessor. Proc. of the design automation conference (1998) 732-737.

[39] D. Burger and T. Austin, “The SimpleScalar Tool Set, Version 2,” Tech. Report No. 1342, Computer Sciences Dept. Univ. of Wisconsin, June 1997.

[40] J. Montanaro, et al; A 190-MHz, 32-b, 0.5W CMOS RISC Microprocessor. IEEE journal of Solid-State circuit 31(11) (Nov 1996) 1703-1714.

[41] S. Segars: ARM7TDMI power consumption. IEEE micro 17(4) (jul/aug 1997) 12-19.

[42] Aristides efthymiou, jim D.Garside, and Steve Temple department of computer science, university of Manchester UK: A comparison power analysis of an asynchronous processor- AN4621 Application note STM32L4 ultra-low-power features overview.

[43] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A Framework for Architectural-Level Power Analysis and Optimizations," Proceedings of the 27th International Symposium on Computer Architecture (ISCA), June 2000.

[44] W. Ye, N. Vijaykrishan, M. Kandemir, and M. J. Irwin, "The Design and Use of SimplePower: A Cycle-Accurate Energy Estimation Tool," Proceedings of the Design Automation Conference, June 2000.

[45] Mikael Hirki: energy and performance profiling of scientific computing, Aalto univ of computer science and engineering, November 20, 2015.

Résumé

L’évolution actuelle des systèmes embarqués tend à leur faire intégrer une puissance de traitement de plus en plus importante tout en conservant, voir en améliorant, leur autonomie énergétique. Si depuis longtemps les techniques de la consommation de diminution ont été recherchées, elles deviennent maintenant primordiales dans l’élaboration d’un système embarqué. Ce mémoire présente un certain nombre de techniques permettant de réduire la consommation énergétique. Certaines solutions sont purement matérielles, d’autres purement logicielles, et d’autres enfin appelées hybrides nécessitent une collaboration entre une partie matérielle et une partie logicielle. Une de ces solutions hybrides est étudiée en détail qui est l’adaptation dynamique de la tension d’alimentation. Cette technique illustre l’impact que peut avoir une technique de diminution de la consommation d’énergie sur un système.

En somme ce résumé nous englobe en général des techniques modernes utilisées pour réduire la consommation en énergie d’un système informatique temps réel monoprocesseur. Il a été montré expérimentalement que lors d’une utilisation intensive du processeur, celui-ci pouvait consommer plus de 50% de l’énergie totale utilisée par le système. Notre objectif principal est d’apporter une nouvelle technique tentant de minimiser la consommation en énergie du système et cependant son exécution.

ملخص

يميل التطور الحالي للأنظمة المدمجة إلى جعلها تدمج قوة المعالجة أكثر وأكثر أهمية مع الحفاظ على استقلالها للطاقة أو تحسينه. إذا كانت أساليب استهلاك الانخفاض مطلوبة لفترة طويلة، فإنها تصبح ضرورية الآن في تطوير نظام مدمج. تقدم هذه المذكرات عددًا من التقنيات لتقليل استهلاك الطاقة. بعض الحلول عبارة عن أجهزة بحتة، وبرامج أخرى بحتة، وآخرون يطلق عليهم الهجين في نهاية المطاف يتطلب التعاون بين جزء من الأجهزة وجزء من البرمجيات. يتم دراسة أحد هذه الحلول المختلطة بالتفصيل وهو التكيف الديناميكي لجهد الإمداد. توضح هذه التقنية التأثير الذي يمكن أن تحدثه تقنية تقليل استهلاك الطاقة على النظام

باختصار، يشمل هذا الملخص بشكل عام التقنيات الحديثة المستخدمة لتقليل استهلاك الطاقة في نظام حاسوب في الوقت الحقيقي للمعالج الواحد. وقد تبين بشكل تجريبي أنه خلال الاستخدام المكثف للمعالج، يمكن أن تستهلك أكثر من 50 ٪ من إجمالي الطاقة المستخدمة من قبل النظام. هدفنا الرئيسي هو تقديم تقنية جديدة في محاولة لتقليل استهلاك الطاقة في النظام وبعد تنفيذه