Contributions à l'étude d'un processeur monolithique 32 bits en technologie CMOS

Abdelaziz Ouerdani

THESE

présentée à

l'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

pour obtenir
le titre de DOCTEUR DE 3ème CYCLE
"Informatique"

par

Abdelaziz OUERDANI

CONTRIBUTIONS A L'ETUDE D'UN PROCESSEUR MONOLITHIQUE 32 BITS EN TECHNOLOGIE CMOS.

Soutenue le 20 juin 1986 devant la commission d'examen

G. MAZARE	Président
F. ANCEAU	
P. GENTIL	Examinateurs
J.L. LARDY	
A. GUYOT	

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Année universitaire 1982-1983

Président de l'Université : D. BLOCH

Vice-Président : René CARRE
Hervé CHERADAME
Marcel IVANES

PROFESSEURS DES UNIVERSITES :

ANCEAU François	E.N.S.I.M.A.G.
BARRAUD Alain	E.N.S.I.E.G.
BAUDELET Bernard	E.N.S.I.E.G.
BESSON Jean	E.N.S.E.E.G.
BLIMAN Samuel	E.N.S.E.R.G.
BLOCH Daniel	E.N.S.I.E.G.
BOIS Philippe	E.N.S.H.G.
BONNETAIN Lucien	E.N.S.E.E.G.
BONNIER Etienne	E.N.S.E.E.G.
BOUVARD Maurice	E.N.S.H.G.
BRISSONNEAU Pierre	E.N.S.I.E.G.
BUYLE BODIN Maurice	E.N.S.E.R.G.
CAVAIGNAC Jean-François	E.N.S.I.E.G.
CHARTIER Germain	E.N.S.I.E.G.
CHENEVIER Pierre	E.N.S.E.R.G.
CHERADAME Hervé	U.E.R.M.C.P.P.
CHERUY Arlette	E.N.S.I.E.G.
CHIAVERINA Jean	U.E.R.M.C.P.P.
COHEN Joseph	E.N.S.E.R.G.
COUMES André	E.N.S.E.R.G.
DURAND Francis	E.N.S.E.E.G.
DURAND Jean-Louis	E.N.S.I.E.G.
FELICI Noël	E.N.S.I.E.G.
FOULARD Claude	E.N.S.I.E.G.
GENTIL Pierre	E.N.S.E.R.G.
GUERIN Bernard	E.N.S.E.R.G.
GUYOT Pierre	E.N.S.E.E.G.
IVANES Marcel	E.N.S.I.E.G.
JAUSSAUD Pierre	E.N.S.I.E.G.
JOUBERT Jean-Claude	E.N.S.I.E.G.
JOURDAIN Geneviève	E.N.S.I.E.G.
LACOUME Jean-Louis	E.N.S.I.E.G.
LATOMBE Jean-Claude	E.N.S.I.M.A.G.

.../...

LESSIEUR Marcel	E.N.S.H.G.
LESPINARD Georges	E.N.S.H.G.
LONGEQUEUE Jean-Pierre	E.N.S.I.E.G.
MAZARE Guy	E.N.S.I.M.A.G.
MOREAU René	E.N.S.H.G.
MORET Roger	E.N.S.I.E.G.
MOSSIERE Jacques	E.N.S.I.M.A.G.
PARIAUD Jean-Charles	E.N.S.E.E.G.
PAUTHENET René	E.N.S.I.E.G.
PERRET René	E.N.S.I.E.G.
PERRET Robert	E.N.S.I.E.G.
PIAU Jean-Michel	E.N.S.H.G.
POLOUJADOFF Michel	E.N.S.I.E.G.
POUPOT Christian	E.N.S.E.R.G.
RAMEAU Jean-Jacques	E.N.S.E.E.G.
RENAUD Maurice	U.E.R.M.C.P.P.
ROBERT André	U.E.R.M.C.P.P.
ROBERT François	E.N.S.I.M.A.G.
SABONNADIERE Jean-Claude	E.N.S.I.E.G.
SAUCIER Gabrielle	E.N.S.I.M.A.G.
SCHLENKER Claire	E.N.S.I.E.G.
SCHLENKER Michel	E.N.S.I.E.G.
SERMET Pierre	E.N.S.E.R.G.
SILVY Jacques	U.E.R.M.C.P.P.
SOHM Jean-Claude	E.N.S.E.E.G.
SOUQUET Jean-Louis	E.N.S.E.E.G.
VEILLON Gérard	E.N.S.I.M.A.G.
ZADWORNY François	E.N.S.E.R.G.

PROFESSEURS ASSOCIES

BASTIN Georges	E.N.S.H.G.
BERRIL John	E.N.S.H.G.
CARREAU Pierre	E.N.S.H.G.
GANDINI Alessandro	U.E.R.M.C.P.P.
HAYASHI Hirashi	E.N.S.I.E.G.

PROFESSEURS UNIVERSITE DES SCIENCES SOCIALES (Grenoble II)

BOLLIET Louis
Chatelin Françoise

PROFESSEURS E.N.S. Mines de Saint-Etienne

RIEU Jean
SOUSTELLE Michel

CHERCHEURS DU C.N.R.S.

FRUCHART Robert	Directeur de Recherche
VACHAUD Georges	Directeur de Recherche

.../...

ALLIBERT Michel	Maître de Recherche
ANSARA Ibrahim	Maître de Recherche
ARMAND Michel	Maître de Recherche
BINDER Gilbert	
CARRE René	Maître de Recherche
DAVID René	Maître de Recherche
DEPORTES Jacques	
DRIOLE Jean	Maître de Recherche
GIGNOUX Damien	
GIVORD Dominique	
GUELIN Pierre	
HOPFINGER Emil	Maître de Recherche
JOUD Jean-Charles	Maître de Recherche
KAMARINOS Georges	Maître de Recherche
KLEITZ Michel	Maître de Recherche
LANDAU Ioan-Dore	Maître de Recherche
LASJAUNIAS J.C.	
MERMET Jean	Maître de Recherche
MUNIER Jacques	Maître de Recherche
PIAU Monique	
PORTESEIL Jean-Louis	
THOLENCE Jean-Louis	
VERDILLON André	

CHERCHEURS du MINISTERE de la RECHERCHE et de la TECHNOLOGIRE (Directeurs et Maîtres de Recherches, ENS Mines de St. Etienne)

LESBATS Pierre	Directeur de Recherche
BISCONDI Michel	Maître de Recherche
KOBYLANSKI André	Maître de Recherche
LE COZE Jean	Maître de Recherche
LALAUZE René	Maître de Recherche
LANCELOT Francis	Maître de Recherche
THEVENOT François	Maître de Recherche
TRAN MINH Canh	Maître de Recherche

PERSONNALITES HABILITEES à DIRIGER des TRAVAUX de RECHERCHE (Décision du Conseil Scientifique)

ALLIBERT Colette	E.N.S.E.E.G.
BERNARD Claude	E.N.S.E.E.G.
BONNET Rolland	E.N.S.E.E.G.
CAILLET Marcel	E.N.S.E.E.G.
CHATILLON Catherine	E.N.S.E.E.G.
CHATILLON Christian	E.N.S.E.E.G.
COULON Michel	E.N.S.E.E.G.
DIARD Jean-Paul	E.N.S.E.E.G.
EUSTAPOPOULOS Nicolas	E.N.S.E.E.G.
FOSTER Panayotis	E.N.S.E.E.G.

.../...

GALERIE Alain	E.N.S.E.E.G.
HAMMOU Abdelkader	E.N.S.E.E.G.
MALMEJAC Yves	E.N.S.E.E.G. (CENG)
MARTIN GARIN Régina	E.N.S.E.E.G.
NGUYEN TRUONG Bernadette	E.N.S.E.E.G.
RAVAINE Denis	E.N.S.E.E.G.
SAINFORT	E.N.S.E.E.G. (CENG)
SARRAZIN Pierre	E.N.S.E.E.G.
SIMON Jean-Paul	E.N.S.E.E.G.
TOUZAIN Philippe	E.N.S.E.E.G.
URBAIN Georges	E.N.S.E.E.G. (Laboratoire des ultra-réfractaires ODEILLON)
GUILHOT Bernard	E.N.S. Mines Saint Etienne
THOMAS Gérard	E.N.S. Mines Saint Etienne
DRIVER Julien	E.N.S. Mines Saint Etienne
BARIBAUD Michel	E.N.S.E.R.G.
BOREL Joseph	E.N.S.E.R.G.
CHOVET Alain	E.N.S.E.R.G.
CHEHIKIAN Alain	E.N.S.E.R.G.
DOLMAZON Jean-Marc	E.N.S.E.R.G.
HERAULT Jeanny	E.N.S.E.R.G.
MONLLOR Christian	E.N.S.E.R.G.
BORNARD Guy	E.N.S.I.E.G.
DESCHIZEAU Pierre	E.N.S.I.E.G.
GLANGEAUD François	E.N.S.I.E.G.
KOFMAN Walter	E.N.S.I.E.G.
LEJEUNE Gérard	E.N.S.I.E.G.
MAZUER Jean	E.N.S.I.E.G.
PERARD Jacques	E.N.S.I.E.G.
REINISCH Raymond	E.N.S.I.E.G.
ALEMANY Antoine	E.N.S.H.G.
BOIS Daniel	E.N.S.H.G.
DARVE Félix	E.N.S.H.G.
MICHEL Jean-Marie	E.N.S.H.G.
OBLED Charles	E.N.S.H.G.
ROWE Alain	E.N.S.H.G.
VAUCLIN Michel	E.N.S.H.G.
WACK Bernard	E.N.S.H.G.
BERT Didier	E.N.S.I.M.A.G.
CALMET Jacques	E.N.S.I.M.A.G.
COURTIN Jacques	E.N.S.I.M.A.G.
COURTOIS Bernard	E.N.S.I.M.A.G.
DELLA DORA Jean	E.N.S.I.M.A.G.
FONLUPT Jean	E.N.S.I.M.A.G.
SIFAKIS Joseph	E.N.S.I.M.A.G.
CHARUEL Robert	U.E.R.M.C.P.P.
CADET Jean	C.E.N.G.
COEURE Philippe	C.E.N.G. (LETI)

.../...

DELHAYE Jean-Marc	C.E.N.G. (STT)
DUPUY Michel	C.E.N.G. (LETI)
JOUVE Hubert	C.E.N.G. (LETI)
NICOLAU Yvan	C.E.N.G. (LETI)
NIFENECKER Hervé	C.E.N.G.
PERROUD Paul	C.E.N.G.
PEUZIN Jean-Claude	C.E.N.G. (LETI)
TAIEB Maurice	C.E.N.G.
VINCENDON Marc	C.E.N.G.

LABORATOIRES EXTERIEURS

DEMOULIN Eric	C.N.E.T.
DEVINE	C.N.E.T. (R.A.B.)
GERBER Roland	C.N.E.T.
MERCKEL Gérard	C.N.E.T.
PAULEAU Yves	C.N.E.T.
GAUBERT C.	I.N.S.A. Lyon

RESUME

La nécessité d'une conception sûre et descendante des circuits intégrés VLSI (Very Large Scale Integration) est largement reconnue. Cette thèse présente une étude sur les propriétés statiques et dynamiques des dessins de masques des principaux blocs constituant un circuit intégré réalisés en technologie CMOS. La méthode proposée est une conception par affinement successifs des spécifications. On distingue deux choix fondamentaux dans la conception des circuits:

- le choix des algorithmes,

- le choix du chemin de données associé aux blocs fonctionnels.

Les validations partielles de conception étant faites par analyse et simulation.

MOTS-CLES:

CI - VLSI - CAO - plan de masse - topologie - methodologie descendante - CMOS - assemblage de cellules - Partie Opérative - algorithme d'interprétation - niveau d'interprétation

a mi chichita Silvia

a mi "petit poussin" Dianita

que lo han vivido conmigo

a mon père

en souvenir de ma mère

a mi viejita y madre Silvia con todo el cariño y reconocimiento

a toda mi familia tan lejana y que guardo muy cerca en mi corazón

AVANT-PROPOS

AVANT-PROPOS

Le présent travail a été entrepris, tout d'abord, au sein de l'équipe de Recherche en Architecture d'Ordinateurs de l'ENSIMAG à Grenoble dirigée par le Professeur François ANCEAU entre Octobre 1983 et fin Juillet 1984, sous un contrat de recherche passé entre l'Association pour le Développement et la Recherche à Grenoble (ADR) et BULL SYSTEMES (Les Clayes sous Bois). La suite de cette recherche est faite au sein de la Compagnie BULL S.A.

Le travail présenté dans cette thèse est le résultat d'une partie de mes contributions a l'étude d'un processeur monolithique 32 bits en technologie CMOS à deux niveaux de métalisation. Mes études, dans le cadre de ce projet ont été orienté tout particulièrement vers les problèmes suivants:

- étude de la réalisation des fonctions de calcul (arithmétique et flottant).

- étude du séquencement des instructions de base et en flottant.

- conception et implantation de différents blocs fonctionnels de la Partie Opérative du processeur en question.

Cette thèse se présente sous la forme de deux grandes parties:

- la partie A contient trois chapitres concernant une introduction générale, les aspects méthodologiques et algorithmiques pour la conception des circuits intégrés VLSI;

- la partie B contient des chapitres décrivant les réalisations apportées.

Cette étude a posé des problèmes d'architecture, de conception des circuits intégrés très complexes et d'algorithmique des fonctions de calcul.

La mise en route de ce sujet a demandé l'acquisition d'une masse très importante d'informations relatives à la machine que nous prenons en exemple.

CONTRIBUTIONS A L'ETUDE D'UN PROCESSEUR MONOLITHIQUE 32 BITS EN TECHNOLOGIE CMOS

On n'a pas besoin de lumière, pour voir clair

PICASSO

L'enterrement du Comte d'Orgaz

REMERCIEMENTS

Mes plus vifs remerciements vont à Messieurs les membres du Jury de cette thèse :

- à Monsieur Guy MAZARE, Professeur à l'ENSIMAG/INPG, qui me fait l'honneur de présider ce Jury et envers lequel je garde un bon souvenir de son cours de CAO et de mon début de thèse au CNET/Meylan. Qu'il soit assuré de mes meilleures sympathies.

- à Monsieur François ANCEAU, ex-professeur à l'ENSIMAG/INPG, chef de projet à BULL SYSTEMES, à qui je dois beaucoup pour m'avoir accordé sa confiance, fait partagé sa passion pour l'architecture des ordinateurs et "poussé" à participer à un projet d'une ampleur telle qu'il ne pouvait être réalisé qu'en milieu industriel.

- à Monsieur Pierre GENTIL, Professeur à l'ENSERG/INPG, Directeur du Centre Interuniversitaire de Microélectronique (CIM), qui a accépté de participer au Jury de cette thèse et avec lequel j'ai travaillé sur les TD de son cours de microélectronique à l'ENSIMAG.

- à Monsieur Jean Louis LARDY, Ingénieur, Responsable de l'Equipe de Conception au CNET/Meylan, pour le jugement qu'il a bien voulu porter sur cette étude.

- à Monsieur Alain GUYOT, maître-assistant à l'ENSIMAG/INPG, pour les nombreuses discussions qui ont permis d'enrichir le contenu de cette thèse.

Cette thèse est le fruit d'un travail personnel de recherches et de rédaction, elle n'aurait cependant pas vu le jour sans le concours de :

- tous les membres de l'Equipe de Recherche en Architecture des Ordinateurs, en particulier, Emile BOURSIER, avec qui j'ai partagé la passion des micro-ordinateurs.

- BULL SYSTEMES, Direction Etudes et Développements Matériel (DEDM), pour m'avoir permis de rédiger et soutenir cette thèse.

- tous mes collègues de BULL S.A, pour leur soutien et leur encouragement.

- Monsieur D. IGLESIAS et le service de reprographie du Laboratoire IMAG, pour leur sérieux et leur éfficacité dans le tirage de ce document.

TABLE DES MATIERES

-/-/-/-/-/-

PARTIE A: LES METHODOLOGIES DE CONCEPTION DES CI-VLSI

****

-/-/-/-/-/-

PARTIE B: LES REALISATIONS

****

****

-/-/-/-/-/-

-/-/-/-/-/-

P A R T I E A

LES METHODOLOGIES DE CONCEPTION DES CI - VLSI

I - INTRODUCTION

L'appellation microélectronique, littéralement électronique du micron, correspond à l'ordre de grandeur des géométries qu'il faut savoir graver sur un même matériau, généralement du silicium, pour créer et fabriquer un circuit intégré complexe.

Un circuit intégré (ou CI) est donc un assemblage -astucieux- regroupant un nombre important de composants électroniques actifs (transistors, diodes, etc..) fabriqués et produits simultanément au cours d'un processus unique. Ces composants sont interconnectés entre eux sur une même pastille de silicium monocristallin de quelques millimètres carrés de surface.

Après le processus de fabrication, cette partie active du silicium, ou "puce", est encapsulée dans un boîtier plastique, céramique ou métallique.

Les circuits intégrés sont apparus dans la deuxième moitié de la décenie 60 (plus tôt dans les machines militaires) à partir des brevets déposés par les sociétés américaines Texas Instruments et Fairchild; ces C.I ont constitués la grande révolution électronique, celle que nous voyons se dérouler de façon continue depuis cette époque, à un rythme qui ne faiblit pas: des composants électroniques de plus en plus petits, réalisant d'avantages de fonctions électroniques de plus en plus complexes à des vitesses toujours plus grandes et pour un coût toujours plus bas comme le montre la figure I.1. La complexité des CI successifs est bien sûr le premier facteur responsable de cette baisse.

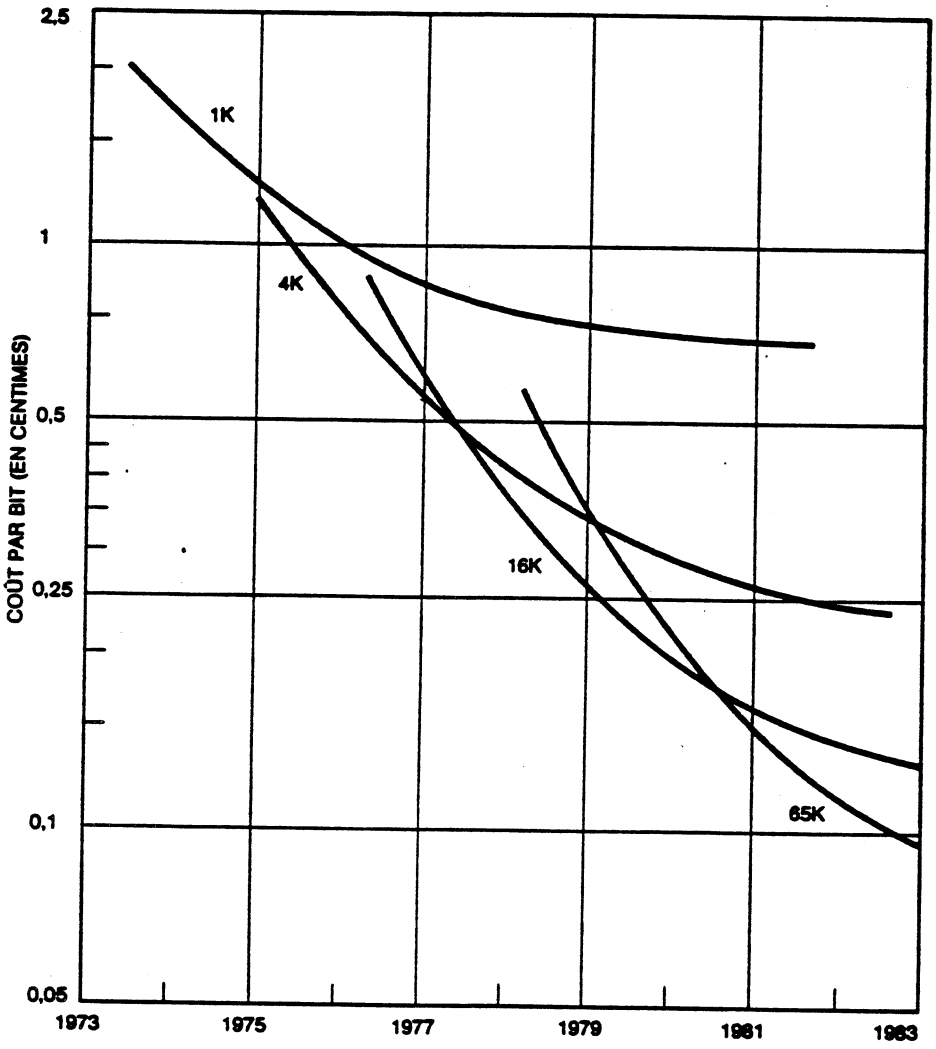


Figure I.1: évolution des coûts par bit des mémoires d'ordinateurs (SA-77)

La densité d'intégration de ces éléments actifs s'est accrue dans le temps en passant par les étapes de l'intégration simple (SSI ou Single Scale Integration), à moyenne échelle (MSI ou Middle Scale Integration), à grande (LSI ou Large Scale Integration) puis à très grande échelle (VLSI ou Very Large Scale Integration) et tous les indices industriels prévoient que l'accroissement des densités d'intégration ainsi que la diminution des temps de commutation des portes élémentaires se poursuivra d'une façon continue au moins jusqu'à la fin des années 80.

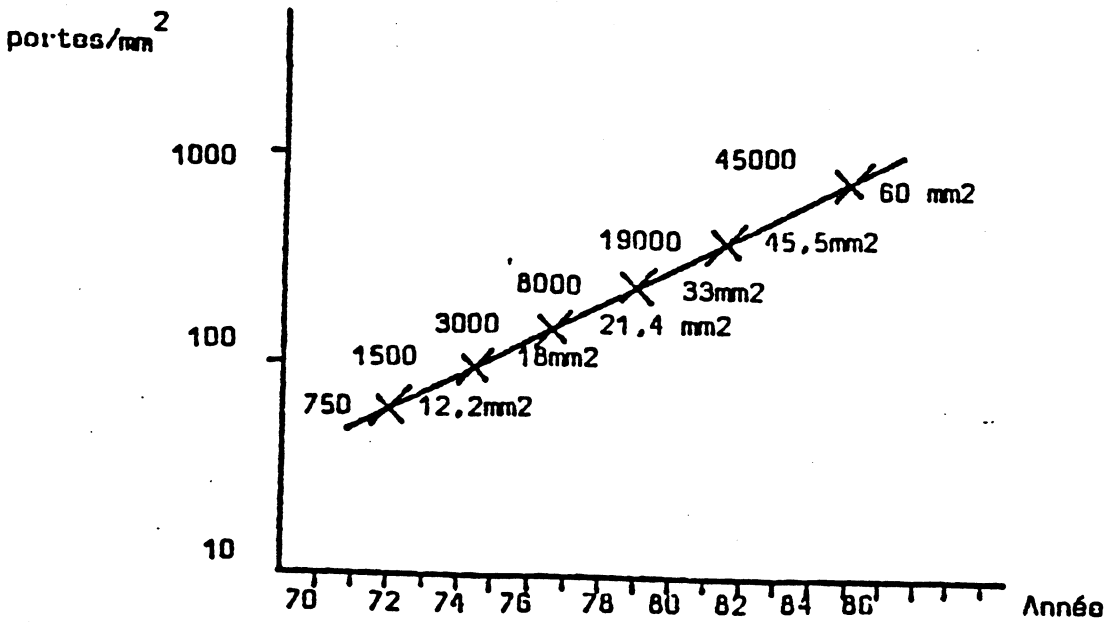


Figure I.2: évolution de la complexité des circuits
(FAG-79)

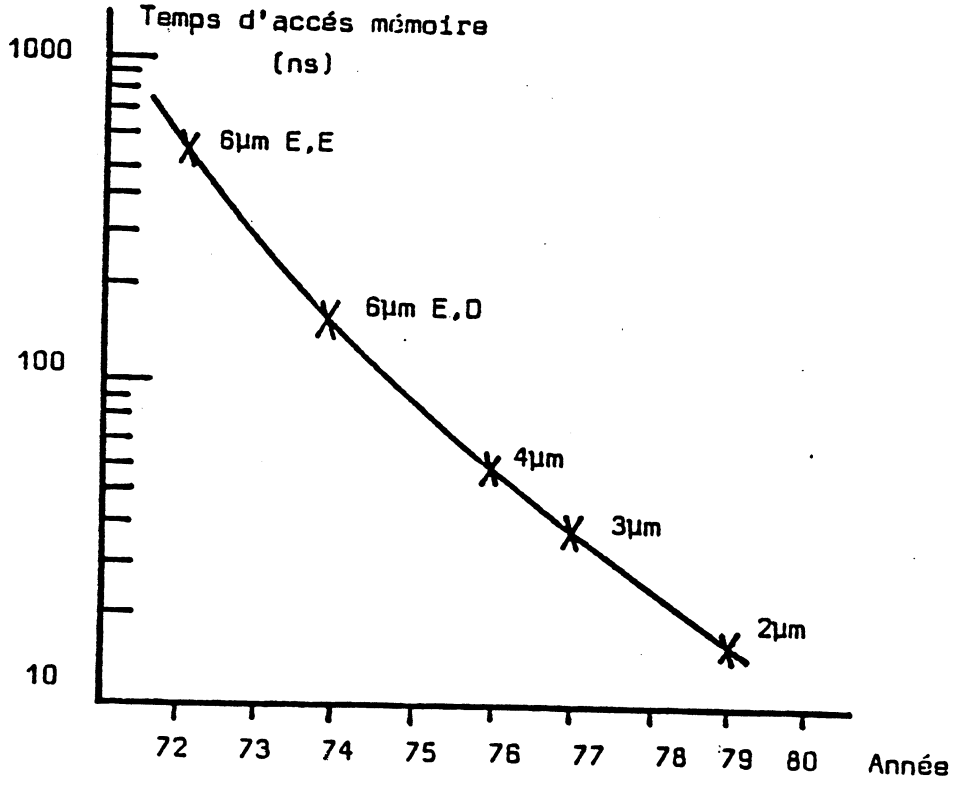


Figure I.3: évolution des temps de réponse des RAM 1kbits
(d'après JEC-79)

Cette complexité n'a été rendue possible que grace à l'affinement de plus en plus spectaculaire de la technologie qui a permis à chaque fois une densité d'intégration toujours plus grande des transistors actifs implantés sur une même tranche de silicium. En effet, les recherches technologiques accroissent le rendement de fabrication de deux manières:

- en réduisant les dimensions intrinsèques des éléments constitutifs, on augmente la densité d'éléments actifs par unité de surface;

- en réduisant les défauts de fabrication, on augmente la surface des circuits qu'il est possible de produire dans des conditions économiquement viables (cf figure I.4).

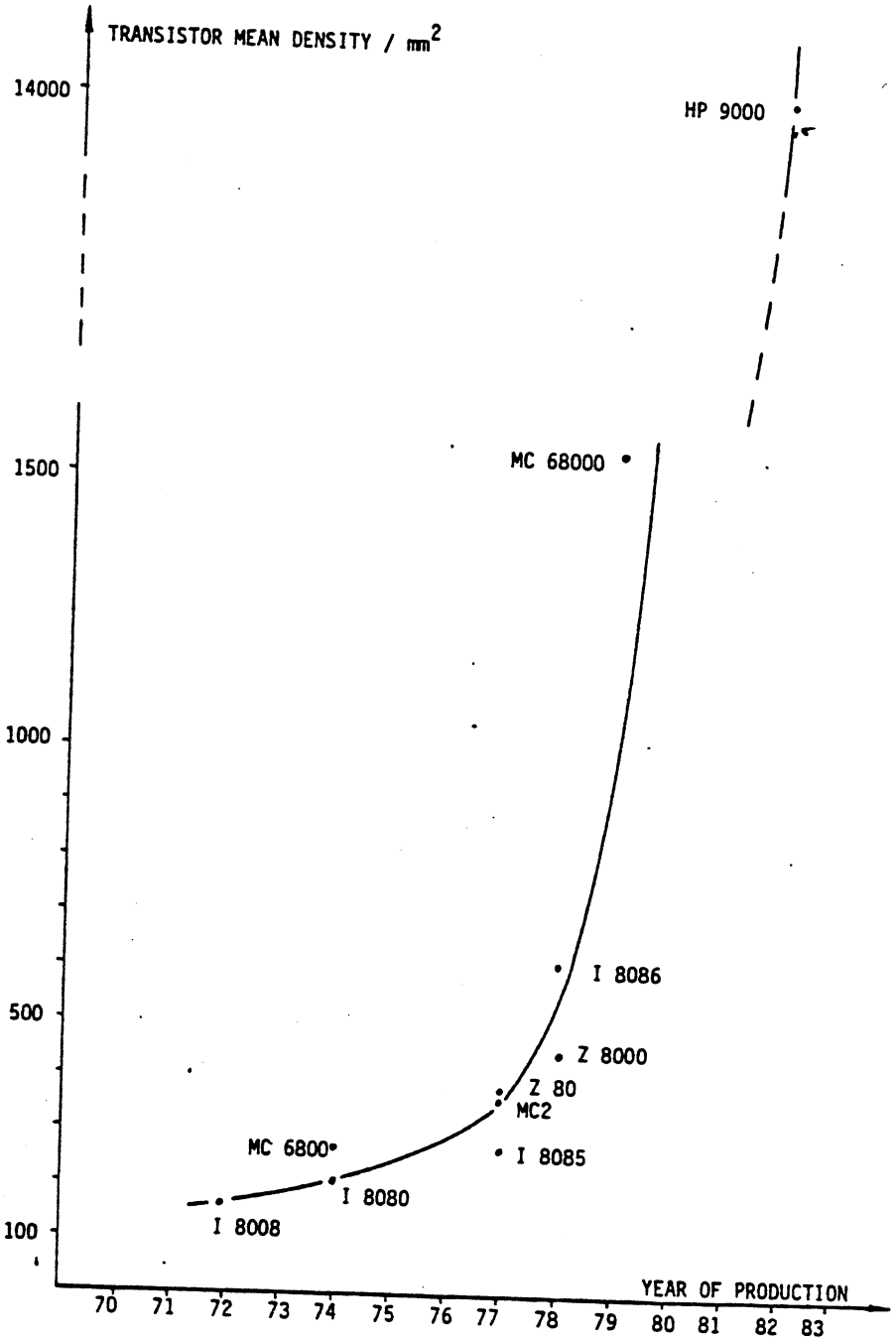


Figure I.4: évolution de la densité moyenne
(d'après LAT-79)

Le nombre de motifs élémentaires, essentiellement les transistors MOS (Metal-Oxide-Semiconductor en anglais), que l'on sait implanter et interconnecter sur un même élément de semiconducteur (silicium) d'environ un demi centimètre carré de surface, va actuellement de quelques dizaines de milliers à quelques centaines de milliers de transistors selon la régularité du circuit.

Pour une grande part, les gains obtenus sont dûs, en particulier, aux progrès continuels de la technologie électronique, soit directement avec l'apparition des nouveaux composants ou de nouvelles techniques, soit indirectement par la baisse des prix ou la modification des coûts entre les divers composants, permettant de repenser la conception des circuits. En particulier, "avec l'accroissement du niveau d'intégration, tous les paramétres caractéristiques des systèmes électroniques ont pu être améliorés simultanément" (réf. FOL) : vitesse fiabilité, énergie dissipée, prix.

Depuis la réalisation des premiers transistors à effet de champs sur substrat de silicium vers la fin des année 50, le nombre des éléments actifs que l'on sait intégrer sur une "puce" ou pastille de silicium (chip en anglais) s'accroît de façon exponentielle, en suivant la "loi de Moore": le nombre de transistors que l'on sait intégrer sur un circuit monolithique double approximativement tous les deux ans.

Cette observation empirique est vérifiée sur les vingts dernières années et tout donne à penser qu'elle le restera pendant au moins dix ans. On est ainsi passé de 1000 transistors en 1970 à 10.000 en 1975 et à 100.000 en 1980 et la mémoire de 1 mégabit existe déjà! Le million de transistors par puce sera réalisé industriellement d'ici la fin des années 1988 car les techniques et les méthodes nécessaires à cette réalisation sont déjà maitrisées dans les laboratoires les plus avancés.

Sur la figure I.5, on voit que la tendance actuelle dans ce domaine est marquée par une croissance continue de la complexité des composants

monolithiques qui se manifeste par une augmentation du nombre de transistors par circuit et une diversification des fonctions à implanter.

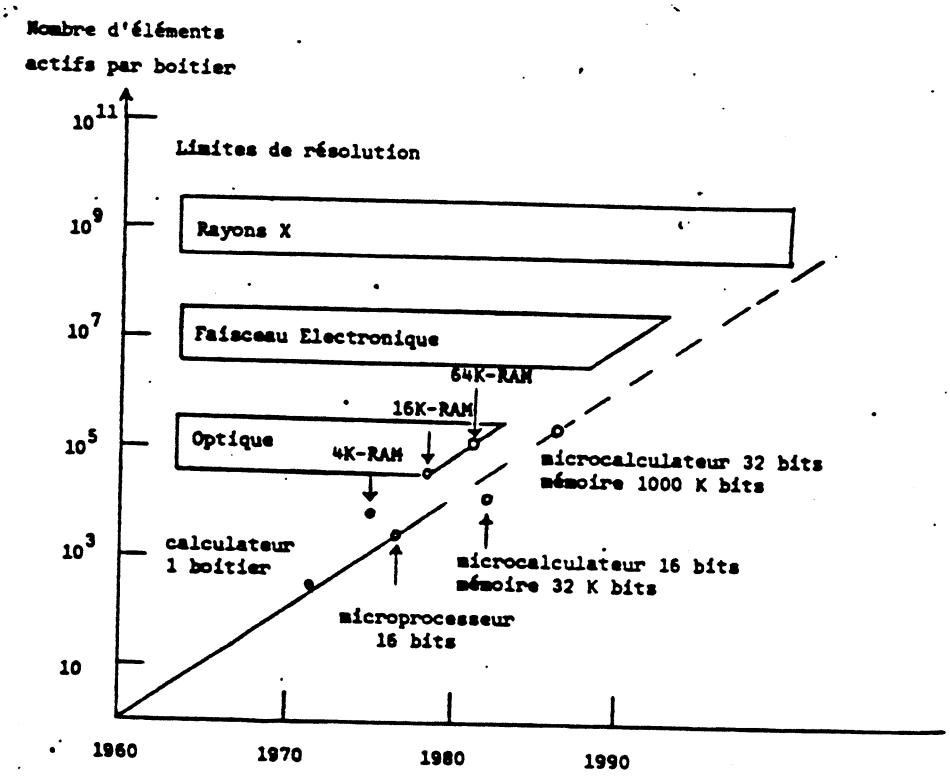


Figure I.5: complexité des boîtiers semi-conducteurs
(diagramme T.I BAN-78)

Le développement de cette densité d'intégration à caractère exponentiel proposée actuellement par les technologues (ou "fondeurs de silicium") n'a été possible que grace à des progrés technologiques constants, permettant d'augmenter les performances des ordinateurs intégrés, ainsi que l'accroissement des méthodes architecturales de plus en plus fines et hiérarchisées mises au point par les concepteurs et architectes d'ordinateurs intégrés et à la maîtrise des outils informatiques d'aide à la conception, ou CAO, (cf figure I.6).

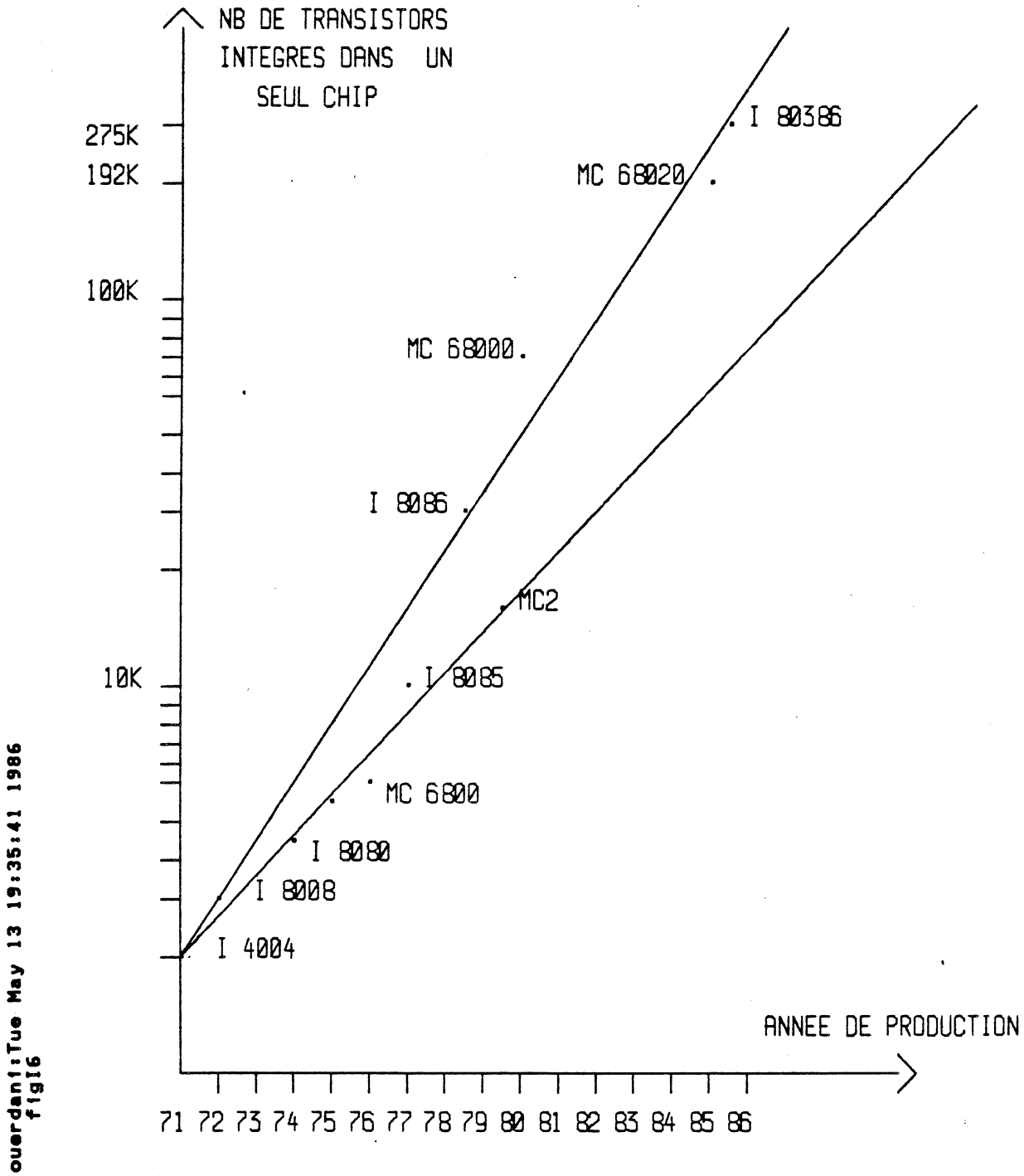


Figure I.6: évolution de nombre total des transistors

Si la loi dégagée par Gordon Moore d'Intel continue à se vérifier dans les années qui viennent, les concepteurs de circuits VLSI vont se

trouver (on y est déjà!) confronté à des défis technologiques de plus en plus difficiles à relever. Les bénéfices obtenus dans cette course à la densité (moindre coût par fonction élémentaire et meilleure fiabilité) se paient par une augmentation des difficultés des réalisateurs qui tiennent essentiellement à l'accroissement du nombre d'éléments et à la complexité fonctionnelle des circuits.

La capacité des ordinateurs à mémoriser, traiter et transcrire l'information dans un temps de plus en plus court s'est considérablement accrue et constitue une révolution en soi. Cette révolution électronique est loin d'être terminée.

Devant la complexité sans cesse croissante des technologies, le concepteur d'ordinateurs intégrés se trouve confronté à de multiples problèmes. En effet, le travail du concepteur des circuits intégrés consiste, à partir des spécifications fonctionnelles et des performances recquises pour le circuit à réaliser:

- à choisir une architecture adaptable à son futur circuit intégré;

- à déterminer le plan de masse de ce circuit, c'est-à-dire à définir le placement et les tailles des différents sous-ensembles du circuit ainsi que leurs interconnexions;

- à concevoir les circuits de base, du point de vue logique et électrique, pouvant réaliser cette architecture;

- à évaluer les dimensions des différents composants qu'il va implanter, après des simulations multiples, pour atteindre les performances désirées;

Après quoi, il faut réaliser l'implantation proprement dite, c'est-à-dire la détermination des géométries de chacun des masques utilisés dans la filière technologique.

Ce développement considérable du nombre des composants intégrables

sur une même pastille de silicium, entraîne pour le travail de conception, deux conséquences:

- Le concepteur de circuits intégrés devient inévitablement de plus en plus architecte, ce qui demande des compétences accrues en microprogrammation.

- Une utilisation de plus en plus importante d'outils informatiques spécialisés d'aide à la conception des C.I (ou CAO) est inévitable et amène souvent le concepteur à les développer soi-même pour sa propre utilisation.

CHAPITRE II

ASPECTS METHODOLOGIQUES POUR LA REALISATION D'UN CI- VLSI

II - ASPECTS METHODOLOGIQUES POUR LA REALISATION D'UN CI-VLSI

II.1. INTRODUCTION

Le chemin qu'il faut parcourir pour fabriquer et mettre sur le marché un circuit intégré est très long. Il existe un gouffre conceptuel entre les spécifications initiales (cahier des charges) du circuit à produire et la puce finale en état de marche dans son boîtier, ce labyrinthe est tel que notre esprit a du mal à l'appréhender.

De nos jours, la complexité sans cesse croissante des circuits intégrés impose une décomposition des étapes de travail, depuis la définition fonctionnelle, en passant par les simulations (logiques, électriques, fonctionnelles etc...) qui valident toutes ces étapes jusqu'au dessin des masques du circuit.

Entre chacune de ces étapes, il est essentiel d'éviter les erreurs dûes à l'action de l'homme sur l'évolution du circuit à produire car c'est là où le choix de la méthode (ou des méthodes) de conception est pris en compte par le concepteur ou l'architecte d'ordinateur intégré. Il s'agit de décisions de haut niveau dont les répercutions sont importantes sur le circuit final et qui engagent fortement la responsabilité du concepteur et l'avenir du circuit intégré à réaliser.

II.2. LES ETAPES DANS LA CONCEPTION ET LA FABRICATION DES CI

Nous rappelons brièvement les grandes étapes dans la conception et la fabrication d'un circuit intégré.

II.2.1. LA CONCEPTION

La conception de circuits VLSI est le processus qui consiste, à partir des spécifications initiales du circuit (cahier des charges),

à le définir et à le décrire complétement, jusqu'à livrer une description "fabricable" sur silicium. Celle-ci est généralement constituée d'une bande magnétique décrivant la géométrie de chacun des masques qui seront utilisés dans la fabrication, bande "formatée" pour piloter directement la machine qui fabrique ces masques. Elle est aussi constituée d'un programme de test qui sera chargé, en sortie de fabrication, de rejeter les circuits défectueux.

Le processus de cette conception peut être décomposé en un certain nombres d'étapes. Les premières sont relativement semblables à celle de la conception des systèmes électroniques classiques: définition architecturale, évaluation puis synthèse logique complète. La différence essentielle réside dans la prise en compte dès ces nivaux des contraintes d'implantation, c'est-à-dire de la disposition des blocs foctionnels et des interconnections sur la surface du silicium, et en particulier de la minimisation nécessaire de leur encombrement.

Le schéma électrique (transistors, capacités, etc....) est ensuite décrit, généralement par cellule; puis le circuit est implanté, c'est-à-dire que les différents transistors et interconnexions sont dessinées sous forme de rectangles ou de polygones associés à chaque niveau de masque. Ceci est réalisé cellule par cellule. Ces cellules sont ensuite juxtaposées et interconnectées pour former de plus grosses cellules, puis des blocs, etc.. jusqu'à constituer le circuit.

II.2.2. L'IMPLANTATION

Parmi les diverses phases de la conception d'un circuit intégré VLSI, celle de l'implantation reste encore la plus critique, et celà à plusieurs titres:

- d'abord parce qu'elle représente un très gros volume de travail, qu'on ne peut pas négliger puisqu'il influe dans le coût total de la conception du circuit;

- ensuite parce que le soin avec lequel est réalisée l'implantation a des conséquences directes et très importantes sur l'efficacité du circuit réalisé, tant du point de vue de sa surface que celui de ses performances (consommation et vitesse);

- et enfin parce que cette phase est très sujette à des erreurs: non respect des régles technologiques de dessin, erreurs de géométries, et surtout erreurs de connexions.

C'est bien pour celà que les systèmes graphiques d'aide à l'implantation, (tels que CALMA, APPLICON, etc..) ont été parmi les premiers outils importants de CAO de circuits intégrés à voir le jour; ils offrent des facilités de stockage des dessins, des manipulations et corrections, mais aussi de vérification des règles de dessin, ainsi que la préparation des sorties pour machines à fabriquer les masques (DAVID MANN, Flacheurs, etc...).

Depuis longtemps déjà, la conception était scindée en plusieurs phases: conception fonctionnelle et logique, simulation fonctionnelle, logique ou électrique selon le niveau de finesse souhaité, tracé du circuit et vérification du dessin. Les efforts les plus anciens apportés par les constructeurs de machine de CAO ont d'ailleurs porté que sur les dernières étapes de la conception (tracé, vérification du dessin) et ce n'est que récemment que les travaux de conception ont retenu l'attention des industriels (Metheus, VTI, SUN, APPLICON, etc ...), avec la mise sur le marché des stations de travail (workstations) pour la conception

Cela ne signifie pas que l'aide à la conception était ignorée auparavant (les grands constructeurs possédaient déjà des outils pour leur propre usage), mais la nouveauté est l'ouverture vers les marchés non captifs, représentés par tous les concepteurs de systèmes qui ressentent le besoin d'élaborer des solutions à base de circuits non standards.

II.2.3. LA FABRICATION

La conception et l'implantation d'un circuit intégré se terminent donc par une spécification informatique de ses plans, c'est-à-dire de la géométrie exacte des diverses couches de matériaux le composent.

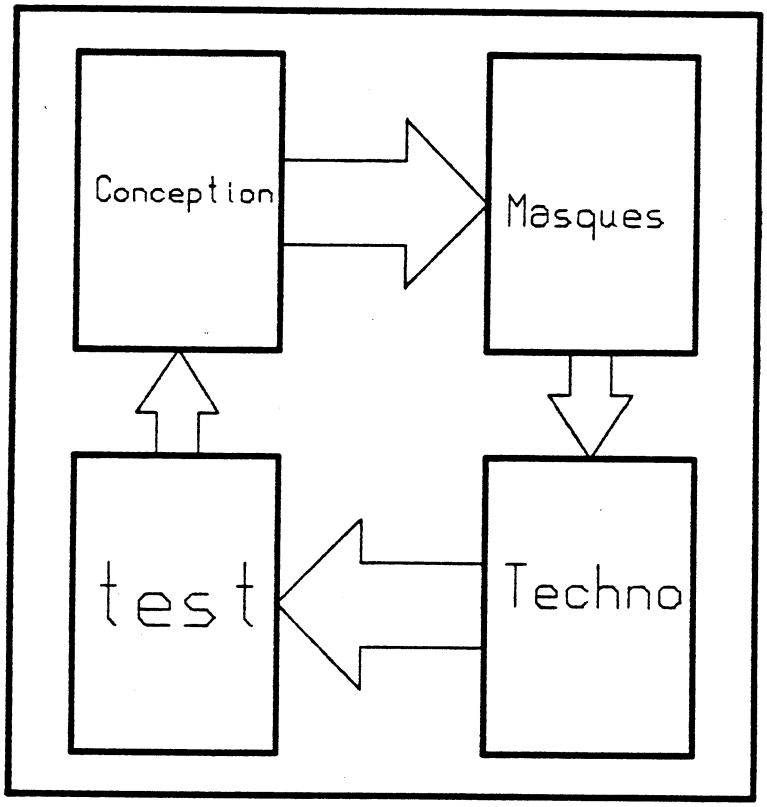
Ces plans sont ensuite matérialisés par une gravure sur une dizaine de plaques de verre, constituant le jeu des masques du circuit.

La fabrication proprement dite part d'un support vierge de silicium monocristallin et va lui faire subir une suite de traitements physico-chimiques. Ces traitements visent à déposer sur le support les diverses "couches" donnant au circuit sa structure électronique propre. Le dépôt minutieux de chaque couche est contrôlé par un seul élément du jeu de masques, qui sert ainsi de "négatif" pour cette étape de traitement.

Au fil de ces traitements, divers tests sont effectués pour s'assurer de la bonne marche de la fabrication, et séparer les circuits défectueux des autres. Comme tout processus physique complexe, la fabrication est entachée d'erreurs d'origines diverses (poussières, radiations, alignements imprécis, etc ...). Il en résulte que seule une faible proportion des circuits fabriqués sont opérationnels. Cette proportion, qu'on appelle le "rendement" de la chaîne technologique, a une incidence économique capitale, et c'est l'un des secrets industriels les mieux gardés. Le rendement décroît d'une façon exponentielle avec la surface des circuits traités, d'où une limite pratique (de l'ordre du cm carré) à la taille des puces réalisables dans une technologie donnée: au delà de cette taille, on s'expose à n'avoir aucun circuit valide en sortie de fabrication.

Après fabrication, les circuits ayant passé les "tests technos" sont montés sur des boîtiers, et renvoyés au concepteur. Celui-ci doit alors tester s'ils répondent ou non aux normes de performances anticipés. On décide alors de revoir la conception, ou de passer à la fabrication en grande série. Nous représentons ces grandes étapes par

la figure II.1 suivante:



ouerdani:Tue May 13 19:11:44 1986
figII1

Figure II.1: cycle conception/fabrication

II.3. STRUCTURATION HIERARCHIQUE DES CIRCUITS

Le plan des masques d'un circuit VLSI, comportant un million de transistors, est décrit par quelque dizaines de millions de vecteurs, dont la représentation en machine nécessite de l'ordre du milliard de bits. L'esprit humain est incapable d'appréhender directement, et à plus forte raison de concevoir sans erreur un objet aussi complexe. Le plus puissant des ordinateurs nécessite des heures de calcul, pour effectuer les traitements les plus simples sur un tel volume d'information.

Il est donc devenu impératif de décomposer un circuit VLSI en un assemblage de modules plus simples, eux même hiérarchiquement

décomposés, et ainsi de suite jusqu'aux modules élémentaires, dont la plus petite taille permet une conception sans erreurs et un traitement informatique efficace. C'est dans cette optique qu'ont été conçu les assembleurs de cellules (ou assembleurs de silicium), comme LUBRICK (ref SCH-85) ou LUCIFER (INRIA).

II.4. LES METHODES DE CONCEPTION

On voit, d'après ce qui est dit plus haut, que la conception d'un circuit complexe à très haute densité d'intégration passe nécessairement par sa décomposition en blocs foctionnels moins complexes, qu'il faut placer et interconnecter; ces blocs eux-mêmes sont composés de blocs plus simples, jusqu'à ce qu'ils soient assez simples pour être implantés directement à la main.

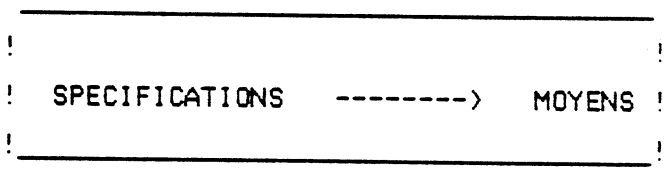
La décomposition en blocs complexes est faite au moment de l'étude architecturale du circuit; les blocs les plus simples sont définis au moment de la conception logique. Mais de ce qui est de l'agencement et de l'interconnection des blocs pour former un bloc plus complexe, il est d'usage de distinguer deux types de méthodes: les méthodes descendantes ("top down") et les méthodes ascendantes ("bottom up").

II.4.1. L'APPROCHE DESCENDANTE

C'est une démarche qui, partant des spécifications initiales (ou cahier des charges) du circuit à réaliser et allant jusqu'à la description finale du produit (implantation micronique) en passant par son architecture adaptée qui permet cette implantation.

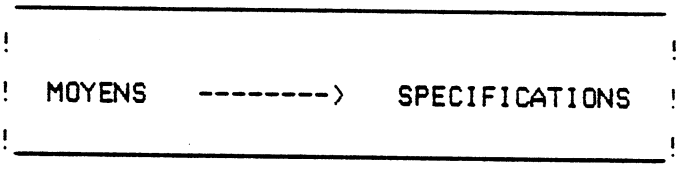
Cette démarche consiste à décider de l'agencement des blocs composants et de leur interconnection dès l'étude architecturale, et avant même que leur contenu ait été implanté. Elle nécessite une bonne expérience du concepteur qui doit évaluer le plus exactement possible l'encombrement des blocs. Elle est basée donc sur la

planification du plan de masse du circuit. La construction prédictive de la surface des blocs fonctionnels qui le constituent. Le plan de masse servira de structure d'évaluation et de gestion pendant la conception du circuit où toutes les modifications effectuées seront reportées.



II.4.2. L'APPROCHE ASCENDANTE

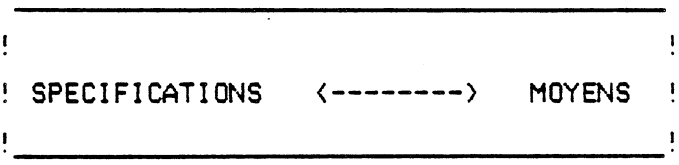
Les méthodes ascendantes consistent au contraire à placer et interconnecter des blocs déjà déssinés (ce sont les moyens dont on dispose) pour construire des blocs plus complexes permettant d'aboutir aux spécifications initiales, mais elles supposent elles aussi une bonne expérience pour que ces blocs aient été judicieusement prévus, et en particulier leurs entrées et leurs sorties soient placées de telle façon qu'elles ne nécessitent pas des interconnexions trop longues.



II.4.3. L'APPROCHE MIXTE

En fait, toute conception passe nécessairement par une phase descendante (pour déterminer le plan de masse du circuit à réaliser et donc de son architecture interne) et une phase ascendante du point de vue implantation des cellules de base pour aboutir à l'architecture; la

distinction entre les deux phases repose plutôt sur le degré de rigidité accordé à l'organisation définie en phase descendante (floor-plan).



Si la décomposition en différentes phases mentionnées plus haut était effectuée naturellement dans le passé, il faut bien admettre que chacune d'elles était relativement indépendante des autres, utilisant ses propres outils de description, les liens entre étapes, la cohérence tout au long de la chaîne étant plus ou moins bien assurée par les relations entre équipes. Aujourd'hui, avec les stations de travail, la cohérence doit être explicitement maintenue de bout en bout: chaque modification dans une étape doit, dans le cas idéal, provoquer les indispensables mises à jour dans les étapes. En particulier, il faut pouvoir assurer la cohérence ultime: celle des spécifications du circuit et celles de son dessin géométrique.

II.5. ARCHITECTURE DE CIRCUITS: EVOLUTION ET OUTILS DE DESCRIPTION

L'architecture des circuits a profondément évolué, s'adaptant ainsi aux possibilités nouvelles offertes par la technologie. Depuis le premier processeur 4 bits en 1972, les microprocesseurs sont passés à 8, 16 et maintenant à 32 bits. En simplifiant grossièrement, on peut dire que, à la taille près, l'architecture logique de ces machines a peu évolué: on retrouve, dans la partie opérative des microprocesseurs 16 bits la structure de base des machines 8 bits; c'est également vrai, à un moindre degré, dans leur partie contrôle.

Qu'en sera-t-il dans l'avenir? Comment vérifier que les centaines de milliers de petits motifs (transistors, connexions, contacts, etc ...) donnent bien finalement le comportement attendu?

Une meilleure solution qui éviterait ce retour en arrière (du dessin vers le schéma électrique) serait de générer automatiquement les connexions à partir des définitions des cellules ou des caractéristiques du système. Une des voies possibles serait le fameux "compilateur de silicium" qui engendre automatiquement le dessin du circuit à partir de ses propriétés, de la même façon qu'un compilateur génère du code objet à partir d'un code source écrit en langage de haut niveau.

II.5.1. SYSTEMES SUR SILICIUM

De plus en plus, aujourd'hui, on intègre sur un seul circuit les fonctions réalisées auparavant par une ou plusieures cartes câblées. En parallèle avec cette intégration sur silicium s'opère une remise en cause des structures: une architecture répondant à divers compromis liés à une réalisation en composants discrets doit être modifiée pour être réalisée de manière monolithique. De fait, la meilleure manière pour réaliser de tels circuits n'est pas de transposer le modèle câblé sur silicium, mais bien de reprendre la conception à partir d'une spécification fonctionnelle de la machine à réaliser.

II.5.1.1. CONCEPTION DES CIRCUITS PAR LES UTILISATEURS

Contrairement aux circuits figurants au catalogue des constructeurs (mémoires, microprocesseurs), de tels circuits spécifiques du domaine d'applications ne sont pas appelés a être fabriqués en nombres massifs: quelques milliers d'exemplaires dont la "valeur ajoutée" vient de l'équipement auquel ils s'intégrent. Le facteur déterminant dans leur conception n'est plus seulement la surface, qui conditionne le coût unitaire de production, mais surtout le temps de mise au point

qui conditionne la date d'arrivée sur le marché du produit fini. Il peut s'agir dans un cas, de remplacer une carte entière d'électronique par un circuit spécifique. Dans un autre cas, l'adjonction d'un circuit périphérique à un microprocesseur permet de gagner un facteur non négligeable dans ses performances.

II.5.2. OUTILS DE CONCEPTION

Une évolution radicale des outils et méthodes d'aide à la conception de circuits est en cours. La raison principale en est la complexité des nouveaux circuits à concevoir, qui rend caduque les méthodes classiquement utilisées. On a constaté que le temps de conception et de mise au point des nouveaux circuits croît lui aussi de façon exponentielle, à technique de conception égale.

Heureusement, les techniques informatiques de conception évoluent rapidement. La baisse des coûts des mémoires, les progrés des microprocesseurs permettent de doter les concepteurs de postes de travail pour la CAO-VLSI. Ces postes sont intéractifs, graphiques et disposent de puissances de calculs et de logiciels comparables à ceux des "gros" ordinateurs du passé.

II.5.2.1. OBJECTIFS DE LA CAO-VLSI

La fabrication, et surtout le test d'un circuit sont des processus longs: trois mois à plus d'un an sont typiquement nécessaires pour faire réaliser les prototypes d'un nouveau circuit . Le temps de test dépend de la nature et du nombre des erreurs de conception; nombreux sont les circuits qui ont été abandonnés à ce stade. L'objectif premier de la CAO-VLSI est de permettre au concepteur de n'envoyer en fabrication que les circuits dont on est, à priori, sûr de son bon fonctionnement. Pour y parvenir, deux méthodes sont envisageables:

II.5.2.2. LA CAO-VLSI CLASSIQUE

Tous les sytèmes de CAO qui sont aujourd'hui opérationnels s'efforcent de faciliter la détection et la correction des erreurs. Celles-ci peuvent aller de la simple erreur de dessin, rectifiable en quelques secondes avec un editeur graphique, à la plus grave des erreurs de conception, dont la correction remet en cause le principe même du fonctionnement du circuit. La détection, et surtout la correction de ces erreurs sur une réalisation matérielle prototype du circuit est d'autant plus coûteuse que le circuit est complexe.

II.5.2.3. LA COMPILATION DE SILICIUM

Pour éviter d'empiler les possibilités d'erreur, l'idéal, pour le concepteur, serait de ne manipuler qu'une seule description de très haut niveau de son circuit. Un ensemble de programmes se chargeraient de "compiler" cette description en un plan de circuit.

Les avantages d'une telle démarche sont multiples:

- élimination des erreurs de bas niveau (géométriques et électriques), dont la coûteuse vérification devient inutile,

- les masques générés peuvent être, dans une certaine mesure, paramétrés par la technologie. Ceci résoud le sérieux problème de "portabilité du silicium", donc l'adaptation à une technologie en constante mutation. De plus, les systèmes réalisés sur silicium deviennent si complexes qu'ils doivent nécessairement être évolutifs: pour en extirper les erreurs de conception et pour suivre les impératifs commerciaux.

- les connaissances préalables requises du concepteur ne sont pas négligeables. Celui-ci peut donc concentrer toute son attention sur les optimisations <u>globales</u> de l'architecture du circuit, au détriment des considérations <u>locales</u> qui sont seules prises en compte dans les outils classique de CAO.

De nombreuses techniques de génération de géométrie, à partir de déscriptions de plus ou moins haut niveau, existent. Citons par exemple APPOLON, ... Aucune ne peut cependant prétendre à l'universalité: sortie de son cadre d'application, chacune de ces méthodes conduit à une génération de circuits beaucoup trop inefficaces pour être d'intérêt général.

CHAPITRE III

ASPECTS ALGORITHMIQUES POUR LA CONCEPTION DE MICROPROCESSEURS

III - ASPECTS ALGORITHMIQUES POUR LA CONCEPTION DE MICROPROCESSEURS

III.1. INTRODUCTION

Les machines informatiques usuelles sont toutes construites pour examiner et interpréter des opérations définies sous la forme de jeu d'instructions.

Un processeur intégré est avant tout une machine informatique séquentielle. Il contient deux ensembles distincts:

- l'ensemble des données ou parties identifiables d'information, qui sont l'objet d'opérations;

- l'ensemble des instructions qui précisent les opérations à effectuer.

Cela signifie que le comportement d'un tel processeur peut être décrit à l'aide d'un (ou de plusieurs) algorithme appelé algorithme d'interprétation des instructions qui spécifient le comportement de cette machine.

Partant de sa description algorithmique, on peut descendre jusqu'à sa réalisation matérielle en appliquant des opérations d'interprétations par affinements successifs selon la méthode descendante; l'algorithme se décompose en deux fonctions: le séquencement et les actions.

La définition comportementale (ou algorithmique) du processeur comporte donc les deux aspects suivants:

--> définition du langage de commandes que doit interpréter le processeur;

--> description des algorithmes internes permettant d'exécuter ces

commandes.

L'analyse de ces deux descriptions débouche sur la définition structurelle de ce processeur, c'est-à-dire la description des éléments physiques utilisés pour la réalisation effective de la machine.

A ces deux parties, on fait correspondre deux éléments disjoints: la partie opérative et la partie contrôle, dissemblables tant au point de vue topologique que fonctionnel.

Un microprocesseur, comme toute machine informatique, est donc défini par son "langage machine", c'est à dire par le jeu d'instructions qu'il est capable d'exécuter. Le choix d'une réalisation matérielle, parmi d'autres, est déterminé par le concepteur durant la phase de définition de l'architecture interne de la machine; cette réalisation concrétise un choix fait parmi de nombreuses implantations possibles succeptibles d'exécuter l'algorithme d'interprétation du jeux d'instruction (réf. ANC-77).

III.2. DESCRIPTION ALGORITHMIQUE

Le comportement de toute machine informatique doit être vue comme la réalisation d'un interpréteur Ii d'un certain langage algorithmique Li (dit langage de définition). La donnée de l'interpréteur se fait en écrivant l'algorithme A qui le décrit, dans un autre langage Ld, dit de description (Ld peut éventuellement être Li). Cet interpréteur lit donc le programme et modifie les données. La quasi totalité des machines exécutent des langages d'instruction. Pour arriver à une description matérielle de la machine, ces langages permettent de décomposer l'interpréteur de programmes en plusieurs interpréteurs d'un certain nombres de langages de plus en plus fins pour arriver à un interpréteur de phases qui donnera la machine physique.

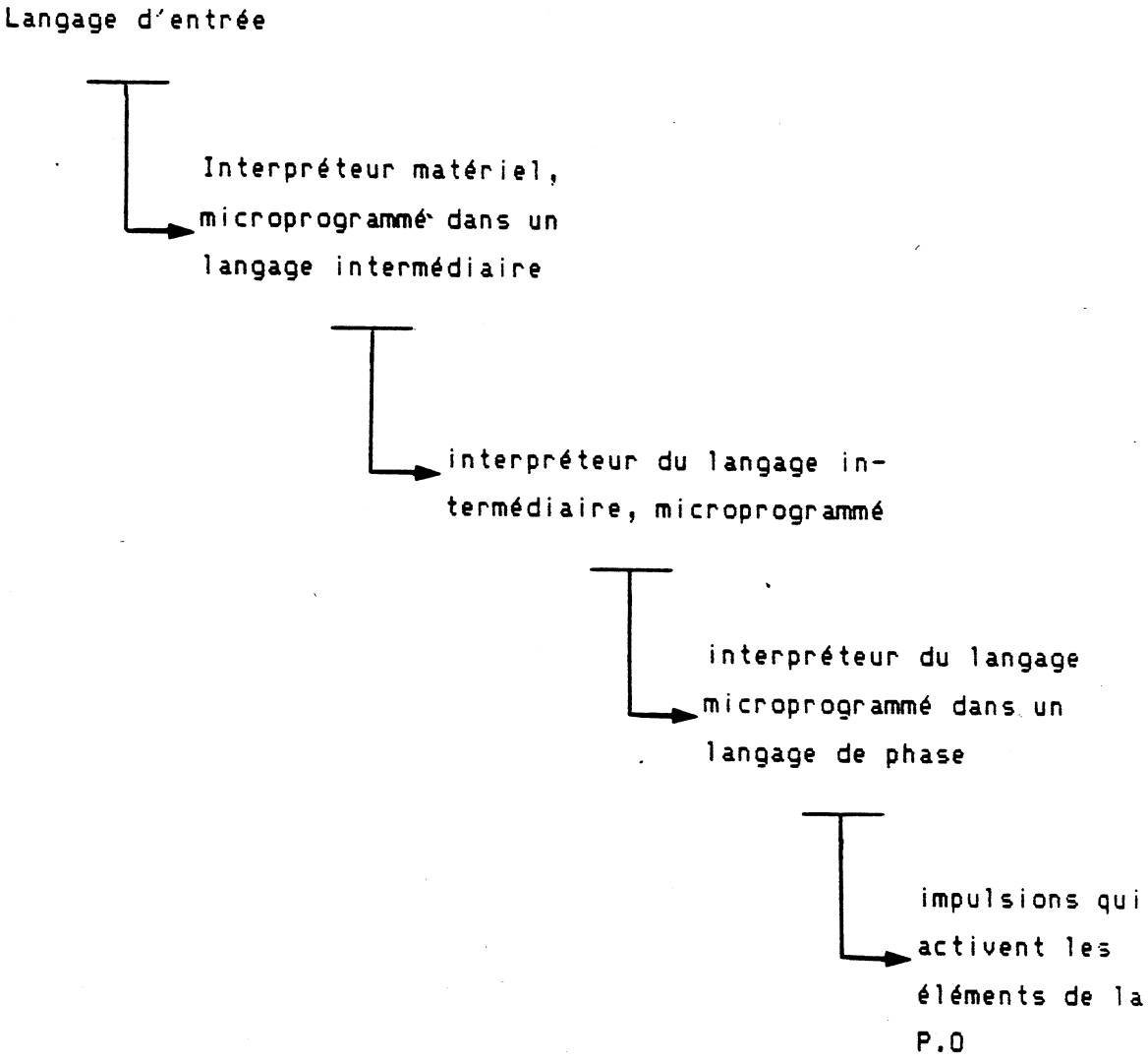


Figure III.1: schéma d'interprétations successives
(MAR-80)

Un algorithme A donné peut avoir différentes formes, aussi bien dans sa réalisation matérielle que dans son expréssion graphique (programmes dans différents langages, organigrammes, etc ...). Les langages par lesquels est exprimé l'algorithme A constituent des ensembles possibles de ressources. Les ressources d'un langage sont dites des primitives alors que les ressources matérielles sont appelées composants.

Notre but est de construire des machines informatiques,

c'est-à-dire, de réaliser matériellement des algorithmes.

Un circuit dont l'algorithme est exprimé par un programme peut être réalisé directement à partir de ce programme. Si les primitives du langage utilisé sont réalisables sur du matériel, alors une transposition directe est possible.

Lorsque les primitives du langage utilisé sont trop complexes et que leur réalisation directe sur du matériel n'est pas possible (où pas intéressante), on procède alors à des descriptions de plus en plus fines, réduisant ainsi, à chaque niveau, la complexité des primitives utilisées. Le dernier niveau atteint ne doit donc avoir que des primitives réalisables sur du matériel. (transfert de registre à registre, opérations appliquées à des variables...).

III.2.1. REALISATION DE L'INTERPRETEUR

Comme nous l'avons évoqué plus haut, les possibilités des organes technologiques élémentaires sont telles qu'il n'est pas possible de réaliser directement l'interpréteur des langages d'instructions complexes. Pour celà nous allons définir des niveaux d'interprétation imbriqués

On définit comme interpréteur Ii du langage Li un algorithme capable d'exécuter tout programme écrit dans ce langage. Le processus d'interprétation est noté:

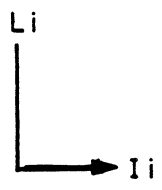


Figure III.2: Interpréteur Ii d'un langage Li

L'interpréteur Ii lui même, peut être programmé dans un langage Li-1. Le processus d'interprétation sera répété, s'il le faut, jusqu'à atteindre le niveau de langage désiré.

On obtient ainsi une chaine dont l'interpréteur de niveau le plus bas I0 peut être réalisé de manière physique. Sa réalisation matérielle en est la machine M capable d'exécuter, indirectement, les instructions de Ln au travers des interprétations imbriquées.

Ainsi l'algorithme d'interprétation des instructions d'un microprocesseur explique la sémantique du jeu d'instruction exécuté. Il est écrit dans un langage dit de définition. Le passage par un où plusieurs interpréteurs successifs autorise une description (et une réalisation) de la machine physique au niveau des phases.

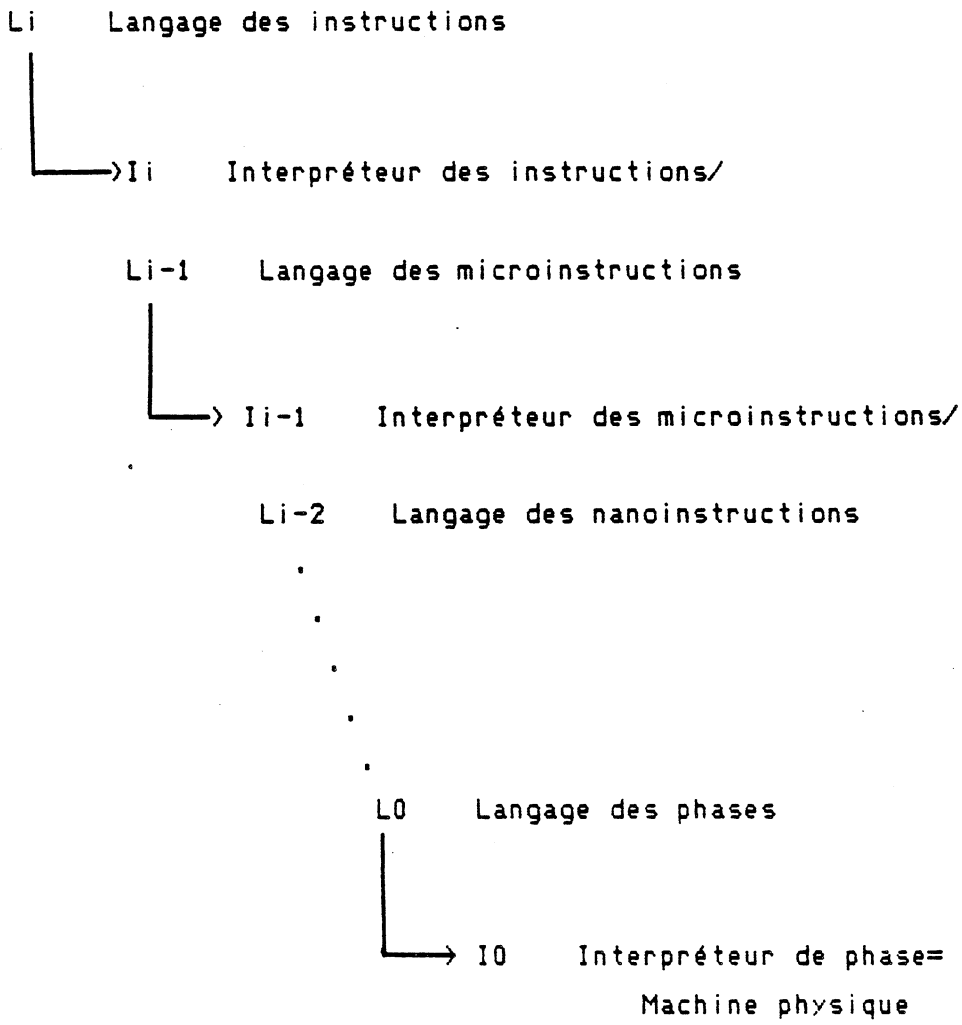


Figure III.3: Exemple de plusieurs niveaux d'interprétation:
l'interpréteur des instructions est écrit dans
le langage des instructions du niveau inférieur
(ANC-77)

A chaque niveau d'interprétation correspond un langage dont les primitives sont plus élémentaires, permettant ainsi une description plus fine des actions.

L0 est un langage de description utilisant des primitivess matérielles, I0 l'interpréteur physique associé. I0 est figé; il n'interprète donc pas tous les programmes écrits en L0 mais seulement un seul.

A chaque niveau d'interprétation correspond une partie opérative mais seule celle de niveau le plus bas existe en tant que structure physique; cette structure dépend des choix pris à des niveaux supérieurs face à la nécessité du compromis entre le coût en matériel et la vitesse désirée. L'exemple classique concerne le choix de la taille d'une UAL (Unité Arithmétique et Logique) par rapport à la taille des données traitées.

Les limitations relatives au matériel sont supposées connues du concepteur; la fonctionnalité et la faisabilité de chaque bloc, tant au point de vue logique qu'électrique nécessite une remise en question permanente de toutes les structures choisies.

III.3. DESCRIPTION STRUCTURELLE

Un microprocesseur peut être définit d'une autre façon. Classiquement, il est composé de deux entités fonctionnelles distinctes: une partie opérative (notée P.O) et une partie contrôle (notée P.C). Cette décomposition est faite, non seuleument du point de vue fonctionnel mais aussi topologique. Ces deux parties coopèrent entre elles en échangeant un certain nombre de signaux: la partie opérative renseignant la partie contrôle de l'état dans lequel elle se trouve tandis que la partie contrôle lui envoie des commandes permettant d'exécuter des opérations élémentaires.

Ces deux définitions, fonctionnelle et structurelle, sont complémentaires, l'une décrit le comportement du processus à réaliser, l'autre sa structure d'implantation.

III.3.1. LA PARTIE OPERATIVE

La P.O est chargée de manipuler et de transformer les données reçues. Elle communique avec le milieu extérieur par des lignes de données correspondant aux entrées/sorties de l'algorithme étudié. Elle recoit des commandes correspondant aux différentes opérations à réaliser, c'est-à-dire:

- aux fonctions devant être réalisées par des organes combinatoires regroupant plusieurs opérations ou prédicats,

- aux sélection d'organes de mémorisation, opérandes des opérations ou prédicats,

- aux chargement d'organes de mémorisation,

- à la sélection d'indices de variables structurées,

- à la fourniture de constante.

Elle émet des informations correspondant à la valeur des différents prédicats calculés.

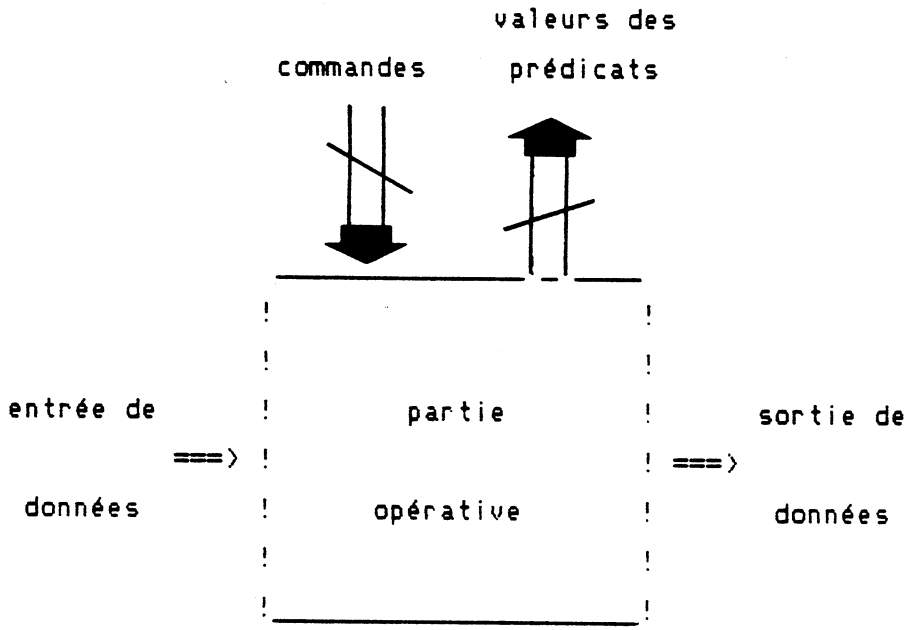


figure III.4: Partie Opérative

III.3.2. LA PARTIE CONTROLE

La partie contrôle a pour rôle de piloter la partie opérative. Elle lui envoie, à chaque cycle, un certain nombre de commandes relatives à l'exécution des instructions. La partie contrôle sera reliée aux monde extérieur par des lignes de synchronisation permettant de coordonner l'activité de tels organes entre eux.

Elle est définie à partir de l'organigramme de l'algorithme d'interprétation; il s'agit, en fait, d'un automate de contrôle qui, d'une part, envoie les commandes vers la partie opérative pour lui indiquer les opérations à effectuer à chaque étape et, d'autre part, assure la progression de l'exécution d'une étape à la suivante.

De plus, pour exécuter le séquencement des instructions, la partie

contrôle à besoin de connaître l'état de la partie opérative; des mots d'état et des prédicats calculés (test) sont générés à cet effet.

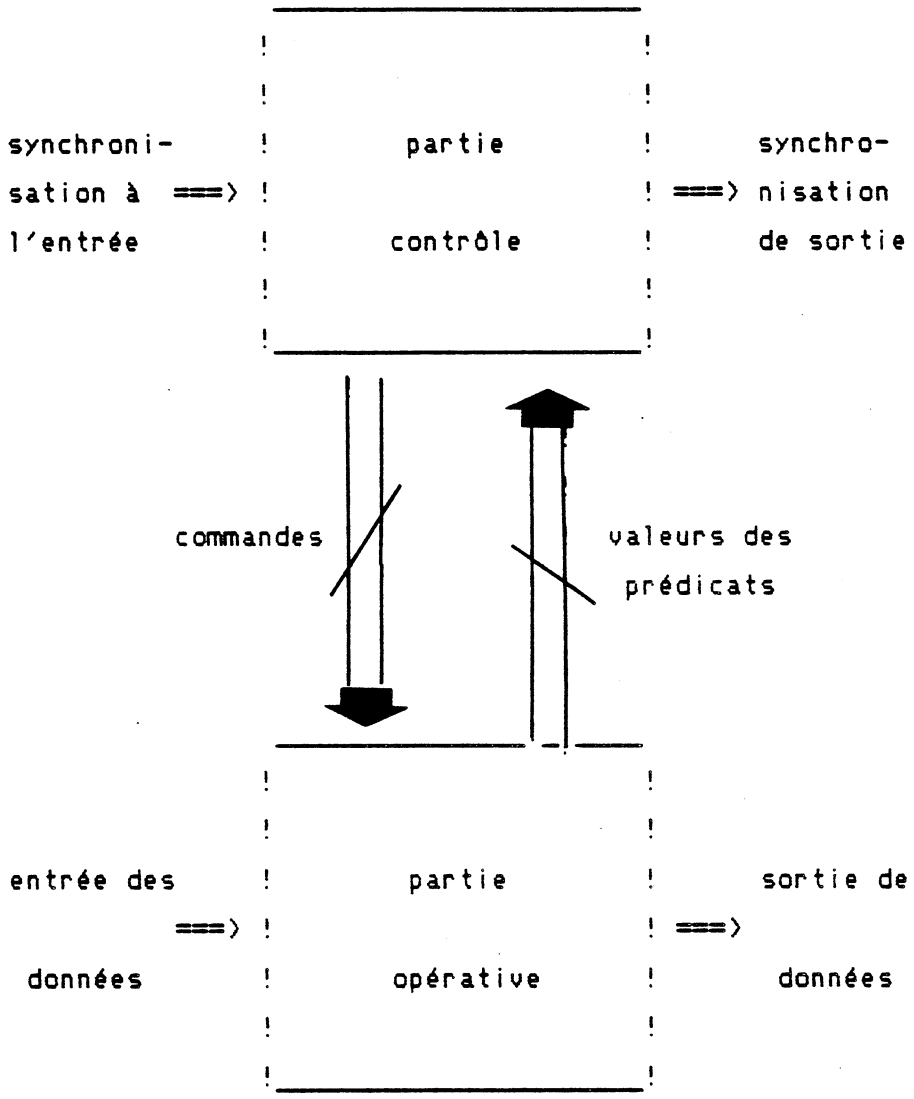


Figure III.5: partie opérative/partie contrôle

Contrairement à la définition de la partie opérative, celle de la partie contrôle ne doit pas forcément être définie à partir de l'interpréteur du niveau le plus bas. En général, la spécification

d'une partie contrôle résulte de l'empilement des spécifications de tous les niveaux de la partie opérative. De plus, a un niveau donné, le même algorithme peut être décrit de différentes façons, impliquant des implémentations matérielles différentes.

III.4. LA MACHINE PHYSIQUE

Les instructions des machines peuvent se répartir en trois classes distinctes vis-à-vis de la complexité de leur séquencement et de leur taux d'apparition dans le temps:

- des instructions dites de base dont le séquencement est simple, très court, très optimisé et très standardisé. La fréquence d'occurence de ces instructions est très élevée,

- des instructions et des fonctions complexes dont le séquencement comporte, à la fois, des séquences optimisées (souvent répétitives) et des séquences complexes (souvent à cause de tests multiples). La fréquence d'occurence de ces instructions est souvent moyenne (0.5 à 1%) ce qui nécessite que leur exécution soit raisonnablement rapide.

- des instructions et des fonctions très complexes dont l'exécution est assurée par des algorithmes longs et complexes manipulant des variables internes à la machine. La fréquence d'occurence de ces instructions est souvent faible ou très faible.

L'organisation physique de la machine doit exploiter ces propriétés de manière à optimiser au maximum le compromis performance/complexité pour permettre la réalisation monolithique du processeur.

III.5. CONCEPTION DU CHEMIN DES DONNEES

La conception de la Partie Opérative d'un processeur est guidée à la fois par des principes généraux exprimés dans les chapitres précédents

et par son cahier des charge:

- il s'agit d'exécuter son jeu d'instruction décrit dans un document de référence;

- d'obtenir la performance maximale, tout en minimisant la surface de silicium nécessaire, en regard de la technologie disponible.

Il faut remarquer qu'il existe des liens très étroits entre les spécifications fonctionnelles (les opérations à réaliser) et les structures matérielles qui sont définies pour les réaliser, ce qui implique un dialogue permanent entre les "logiciens" ou "micro-programmeurs" et les concepteurs ou "dessinateurs" du circuit intégré.

La performance des machines optimisées est en grande partie dûe au recouvrement de l'exécution de deux instructions consécutives: pendant les premiers cycles d'exécution de l'instruction courante, on fait généralement une mise à jour du compteur d'instruction et on prépare le calcul d'adresse pour les opérandes.

P A R T I E B

L E S R E A L I S A T I O N S

CHAPITRE IV

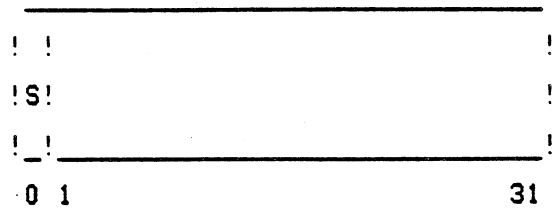
REALISATION DES FONCTIONS DE CALCUL

IV - REALISATION DES FONCTIONS DE CALCUL

La réalisation des instructions issues du cahier des charges d'une machine est conditionnée par leur taux d'occurence et par des considérations de performance (performance en MIPS) que l'on se fixe. Dans cette répartition, on trouve, en particulier, des instructions et des fonctions complexes dont le séquencement présente, soit des séquences répétitives, soit des actions complexes, difficiles à gérer. Dans cette famille d'instructions apparaissent, en particulier, les instructions de multiplication entre des opérandes registres à registres ou entre des opérandes registres et des données venant de la mémoire. Le taux d'occurence de ces instructions est moyen et donc influe sur les performances de la machine. Par conséquent, on est amené à chercher des algorithmes pour que l'exécution de ce type d'instructions soit raisonnablement rapide. Dans cette optique, l'accélération de l'exécution de ces instructions peut être réalisée d'une façon cablée, ce qui nécessite, à priori, l'adjonction de nouveaux matériels dans l'architecture globale de la machine; cette architecture va donc devenir de plus en plus complexe. En effet, l'optimisation des performances de chaque machine est un compromis entre le choix de machines lentes (microprogrammées) d'architecture simple, et des machines rapides (cablées) d'architecture complexe. C'est dans le but d'avoir une machine performante qu'on a cherché à optimiser l'exécution de ces instructions en intégrant des algorithmes rapides tel que l'on va présenter dans la suite:

IV.1. MULTIPLICATION BINAIRE ACCELEREE PAR GROUPEMENT DE BITS

POSITION DU PROBLEME . On veut effectuer la multiplication entre deux nombres binaires signés, écrits dans la représentation en complément à deux et ayant le format 32 bits suivant:



le bit 0 (S) est le bit de signe.

Un nombre binaire signé X s'exprimant dans ce format, a pour valeur:

$$X = x_0 2^{31} + x_1 2^{30} + \ldots + x_{30} 2^1 + x_{31} 2^0$$

Soient A le multiplicande et B le multiplicateur. Alors

$$B = b_0 2^{31} + b_1 2^{30} + \ldots + b_{30} 2^1 + b_{31} 2^0$$

$$A = a_0 2^{31} + a_1 2^{30} + \ldots + a_{30} 2^1 + a_{31} 2^0$$

Le produit (C = A*B) est un nombre binaire signé représenté sur 32 bits (puisqu'on travaille sur 32 bits) et dont la multiplication pure bit à bit nécessite 32 cycles machine. Regardons maintenant comment optimiser cette opération.

ALGORITHME :

Dans la multiplication accélérée, les bits bi, pour i variant de 0 à 31, du multiplicateur B sont groupés par "paquets" de quatre, conformément à l'écriture:

$$B=(-b_0 2^3 + b_1 2^2 + b_2 2^1 + b_3 2^0) 2^{28} + \ldots + (b_{28} 2^3 + b_{29} 2^2 + b_{30} 2^1 + b_{31} 2^0) 2^0$$

Que l'on peut présenter sous la forme compacte suivante:

$$B = k_7 2^{28} + \ldots + k_i 2^{4i} + \ldots + k_0 2^0$$

Chaque coefficient ki prend une valeur comprise entre 0 et 15 pour i variant de 0 à 6; le coefficient k7 contenant le bit de signe varie entre -8 et +7 suivant la valeur de S (1 ou 0).

La multiplication accélérée de A par B donnera le produit C qui est une somme de huit produits partiels (PP)i.

$$C = k_7 A 2^{28} + \ldots + kiA2^{4i} + \ldots + k_0 A 2^0$$

Chaque produit partiel vaut:

$$(PP)i = kiA2^{4i} \text{ pour i variant de 0 à 7}$$

Le passage de (PP)i à (PP)i+1 se fait en multipliant le poids de ce dernier par 2 (base 16). En d'autre termes, le produit partiel d'indice (i+1) est décalé de 4 positions à gauche par rapport au produit partiel d'indice (i), ou, ce qui revient au même, le produit partiel d'indice (i) est décalé de 4 positions à droite par rapport au produit partiel de rang (i+1).

Les bits bi du multiplicateur sont examinés par groupe de quatre. Au cours du processus de la multiplication, ces bits examinés seront les bits du poids faible de B. Il faudrait donc disposer d'un registre à décalage décalant en une seule fois 4 bits vers la droite, dans lequel on mettra les 32 bits de B afin qu'à chaque cycle de décalage, les 4 bits poids faible b_{28}, b_{29}, b_{30}, b_{31} du registre B soient envoyés à une unité de commande pour analyser et déterminer la valeur du coeficient ki par laquelle on multipliera A.

De ce fait, cette multiplication de deux nombres binaires signés de 32 bits va être effectuée en huit cycles machine (au lieu des 32

nécessaires à une multiplication bit à bit); chaque cycle comprendra une phase d'addition (éventuellement sautée) suivie d'une phase de décalage (4 positions à droite). Le temps nécessaire à une multiplication est ainsi divisé par quatre.

Pour que cette méthode puisse marcher, il faut disposer de registres contenant les quantités suivantes:

$$A, 2A, 3A, \ldots\ldots, 14A, 15A.$$

Ceci demanderait un investissement important en matériel et surtout ces quantité ne seront plus représentées dans un format 32 bits! Mais compte tenu de l'égalité $-A=A^*+1$ (A^* étant le complément de A), on peut remarquer que tous ces multiples de A peuvent être obtenus par la décomposition suivante:

A	=	A	
2A	=	2A	
3A	=	4A + A* + 1	ou 2A + A
4A	=	4A	
5A	=	4A + A	
6A	=	4A + 2A	
7A	=	8A + A* + 1	ou 4A + 2A + A
8A	=	8A	
9A	=	8A + A	
10A	=	8A + 2A	
11A	=	16A + 4A* + A* + 2	ou 8A + 2A + A
12A	=	16A + 4A* + 1	ou 8A + 4A
13A	=	16A + 4A* + A + 1	ou 8A + 4A + A
14A	=	16A + 4A* + 2A + 1	ou 8A + 4A + 2A
15A	=	16A + A* + 1	

Il faut donc disposer d'un décaleur de 1, 2 ou 3 positions binaires pour générer à la demande, à partir de A, les multiples 2A, 4A, 8A et

de leurs compléments.

Nous allons utiliser un additionneur à 3 entrées E1, E2 et E3 pour effectuer toutes les sommes des produits partiels nécessaires. Si nous appelons U1, U2 et U3 les registres tampons associés respectivement aux 3 entrées de l'additionneur, alors on pourra générer à partir de U1 les valeurs A, A*, 2A et 0, et à partir de U2 les valeurs 4A, 4A*, 8A, 8A* et 0. Il faudrait donc un décaleur pour chaque tampon d'entrée U1 et U2, et moyennant un multiplexeur associé à chacune de ces deux entrées, on va pouvoir faire toutes les additions nécessaires à notre multiplication. L'entrée E3 associée à un registre U3 muni d'un autre décaleur (4 positions à droite), permet d'obtenir à la fin de chaque cycle de multiplication le résultat partiel (PP)i/16 et donc les 32 bits de poids fort du résultat.

Nous constatons que pour cette solution, il n'est pas possible d'ajouter en même temps que les multiples de A, cités ci-dessus, la valeur 16A qui apparaît dans les 5 dernières lignes du tableau précédent. On propose donc de la transformer en une retenue qui va être rajoutée au cycle suivant.

Le choix des opérandes aux entrées E1 et E2 ainsi que la valeur de la retenue va être commandé par un automate défini par le tableau 1 et le tableau 2.

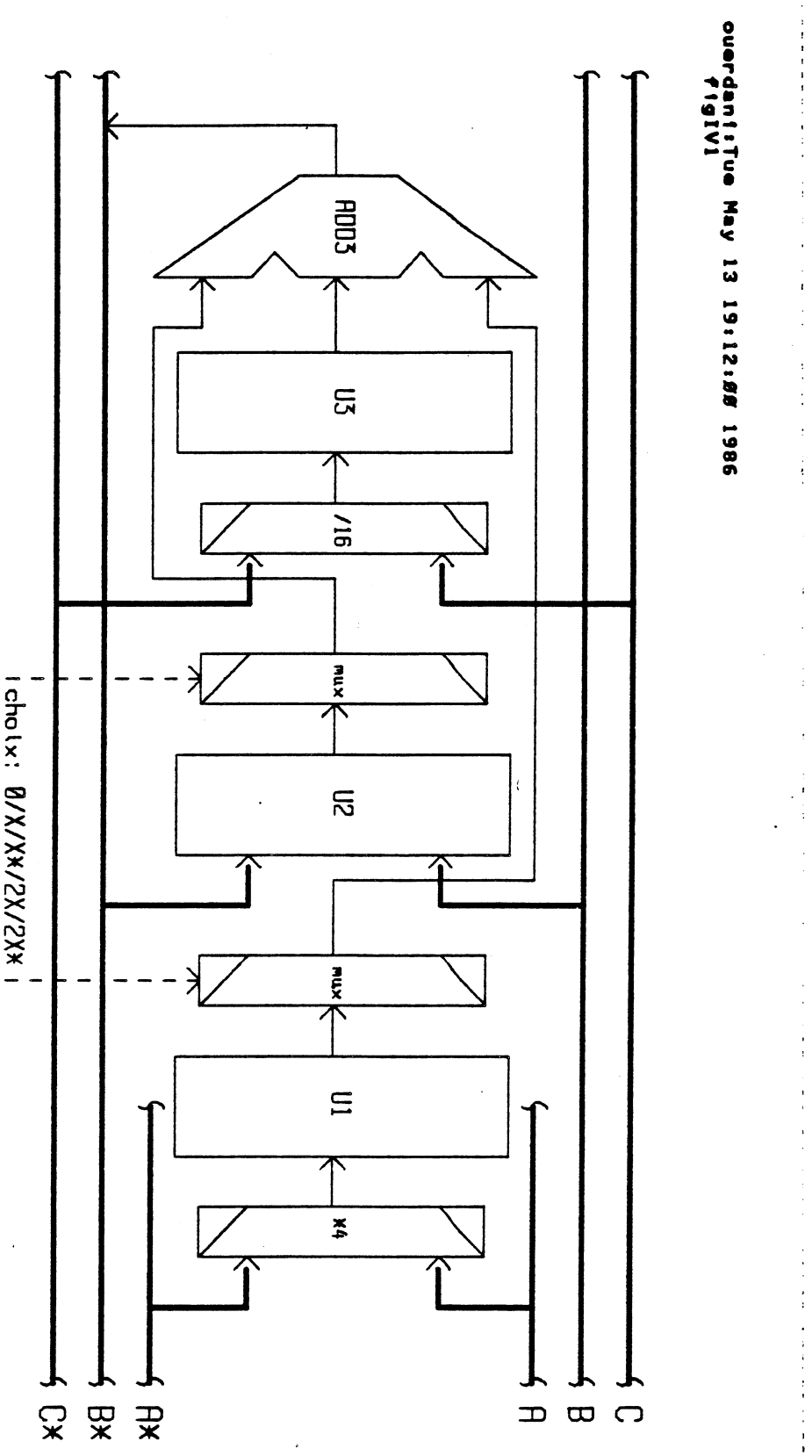


Figure IV.1: Sélection des entrées de A3 pour la multiplicathon

ouerdani:Tue May 13 19:12:14 1986
fig IV2

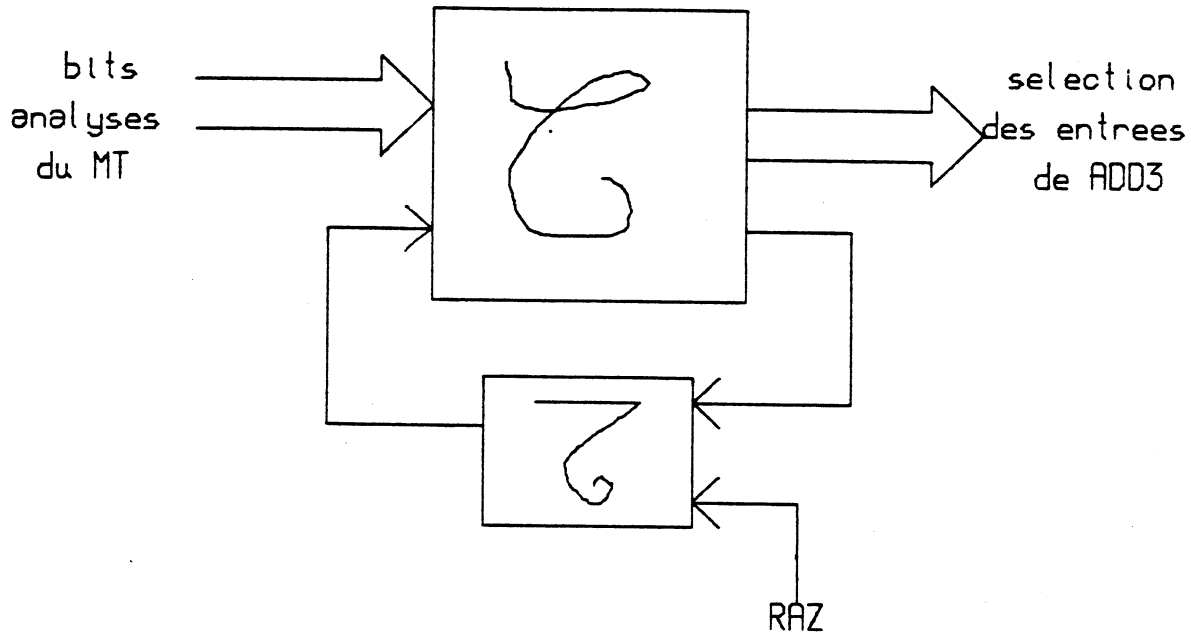


Figure IV.2: automate de multiplication

Pour réaliser à la fois le décalage du multiplicateur et le stockage des 4 bits du résultat poids faible par cycle, on va utiliser un opérateur de décalage que l'on peut noter pour le moment BSH qui permet de faire à la fois des décalages à gauche et des décalage à droite à partir du registre tampon droit BSH.D et du registre tampon gauche BSH.G (respectivement). Nous reviendrons plus loin sur la description détaillée de cet opérateur. Ces décalages vont être de 1, 2, ... ou 32 bits à la fois. Cet opérateur permet de faire aussi des rotations lorsque les deux registres tampons d'entrée sont chargés par la même valeur.

On voit donc bien que si on charge notre multiplicateur dans le tampon d'entrée gauche et le résultat partiel (pour chaque cycle) dans

le tampon d'entrée droit, en décalant à droite de 4 bits le BSH.G on va récupérer ces 4 bits poids faible pour les envoyer au PLA de commande afin de piloter la multiplication au cycle suivant. Les 4 bits poids fort du BSH.G vont être occupés par les 4 bits poids faible du BSH.D. Ces bits représentent les bits du résultat partiel. Donc au bout des huit cycles effectifs de la multiplication on aura accumulé les 32 bits du résultat faible sur le tampon d'entrée BSH.G.

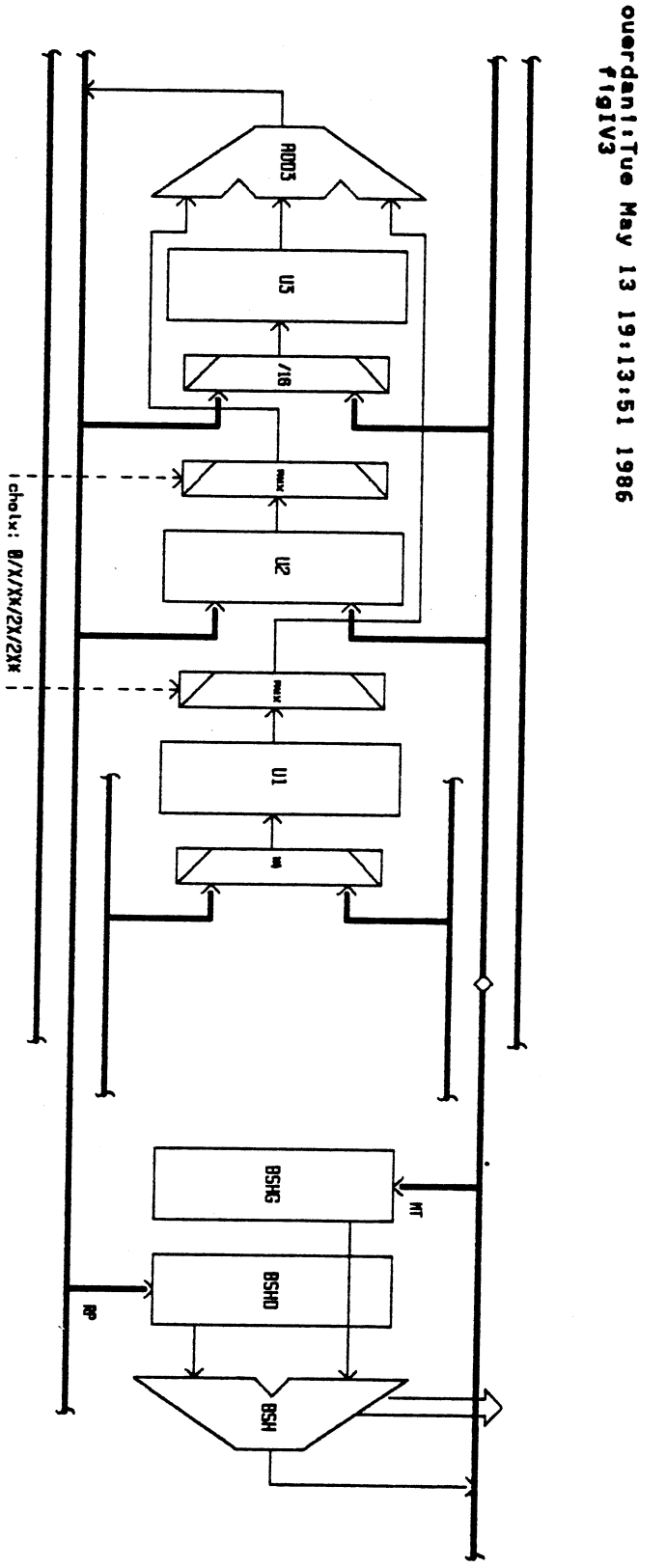


Figure IV.3: Séquencement de la multiplication

Nous commençons donc par charger les registres U1, U2 par le multiplicande A et le registre BSH.G (registre tampon d'entrée du décaleur BSH) par le multiplicateur B. A chaque cycle d'addition, A est multiplié par un groupe de 4 bits de B.

L'analyse de ces 4 bits, au niveau de l'automate de la multiplication, permet la sélection des opérandes sur les entrées E1 et E2 de l'additionneur. Ils sont additionnés avec l'entrée E3 qui reçoit le dernier (PP)i provenant de U3, divisé par 16.

Exemples .

- Si le LSB de U1 est 6 (et l'état de l'automate est à 0) alors :
  E1 = 2.A
  E2 = 4.A (E1 + E2 = 6.A)
  E3 = U3/16 (résultat partiel du calcul précédent)

- Si le LSB de U1 est 10 (et l'état de l'automate est à 1) alors:
  E1 = A* (avec Cin1 = 1)
  E2 = 4A* (avec Cin2 = 1)
  E3 = U3/16
  et 16.A va être ajouté au cycle suivant.

Le résultat de l'addition va être stocké dans U3, les bits du registre BSH.G vont être décalés de quatre positions à droite pour analyser le prochain digit du multiplicateur. Les 4 bits de poids faible du résultat sont stockés dans le digit de poïds fort du registre BSH.G (venus de BSH.D) qui va servir en même temps d'accumulateur. Nous allons arriver ainsi au résultat schématisé par la figure suivante:

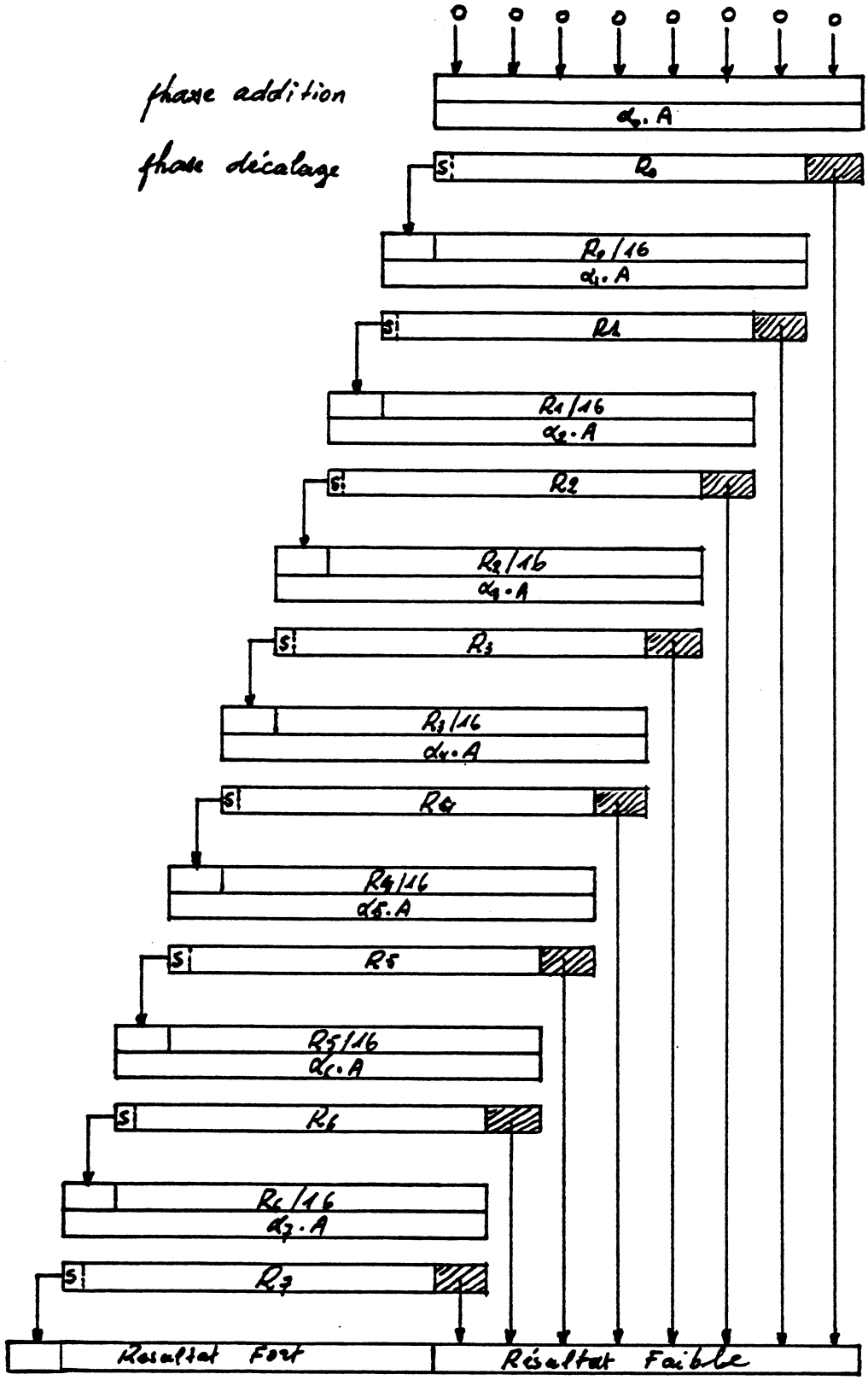


Figure IV.4: phases d'addition/décalage

En fonction des 4 bits de poids faible du multiplicateur, la multiplication se résume donc comme une succession d'addition-décalage suivant l'algorithme des deux tableaux suivants:

bits	cycle C	U1	U2	cycle S	Cin1	Cin2
0 0 0 0	0	0	0	0	0	0
0 0 0 1	0	A	0	0	0	0
0 0 1 0	0	2A	0	0	0	0
0 0 1 1	0	A*	4A	0	1	0
0 1 0 0	0	0	4A	0	0	0
0 1 0 1	0	A	4A	0	0	0
0 1 1 0	0	2A	4A	0	0	0
0 1 1 1	0	A*	8A	0	1	0
1 0 0 0	0	0	8A	0	0	0
1 0 0 1	0	A	8A	0	0	0
1 0 1 0	0	2A	8A	0	0	0
1 0 1 1	0	A*	4A*	1	1	1
1 1 0 0	0	0	4A*	1	1	0
1 1 0 1	0	A	4A*	1	1	0
1 1 1 0	0	2A	4A*	1	1	0
1 1 1 1	0	A*	0	1	1	0
0 0 0 0	1	A	0	0	0	0
0 0 0 1	1	2A	0	0	0	0
0 0 1 0	1	A*	4A	0	1	0
0 0 1 1	1	0	4A	0	0	0
0 1 0 0	1	A	4A	0	0	0
0 1 0 1	1	2A	4A	0	0	0
0 1 1 0	1	A*	8A	0	1	0
0 1 1 1	1	0	8A	0	0	0
1 0 0 0	1	A	8A	0	0	0
1 0 0 1	1	2A	8A	0	0	0
1 0 1 0	1	A*	4A*	1	1	1
1 0 1 1	1	0	4A*	1	1	0
1 1 0 0	1	A	4A*	1	1	0
1 1 0 1	1	2A	4A*	1	1	0
1 1 1 0	1	A*	0	1	1	0
1 1 1 1	1	0	0	1	0	0

Tableau 1

Le cycle C étant le cycle Courant et le cycle S, le cycle suivant.

Au huitième cycle de la multiplication, si k7 est positif il sera traité comme les autres coefficients sinon le PLA commandera l'algorithme suivant:

bits analysés	cycle courant	entrée U1 A,2A,A*,0	entrée U2 4A,8A,8A*,4A*,0	cycle suivant	Cin1	Cin2
k7 négatif						
1 0 0 0	0	0	8A*	0	1	0
1 0 0 1	0	A	8A*	0	1	0
1 0 1 0	0	2A	8A*	0	1	0
1 0 1 1	0	A*	4A*	0	1	1
1 1 0 0	0	0	4A*	0	1	0
1 1 0 1	0	A	4A*	0	1	0
1 1 1 0	0	2A	4A*	0	1	0
1 1 1 1	0	A*	0	0	1	0
1 0 0 0	1	A	8A*	0	1	0
1 0 0 1	1	2A	8A*	0	1	0
1 0 1 0	1	A*	4A*	0	1	1
1 0 1 1	1	0	4A*	0	1	0
1 1 0 0	1	A	4A*	0	1	0
1 1 0 1	1	2A	4A*	0	1	0
1 1 1 0	1	A*	0	0	1	0
1 1 1 1	1	0	0	0	0	0

Tableau 2.

Regardons le déroulement d'exécution d'une multiplication entre deux registres généraux GR1 et GR2. Cette instruction va s'exécuter sur 10 cycles visibles t+0,....,t+9 dont 8 cycles pour le calcul. On multiplie l'opérande du registre GR1 par l'opérande du registre GR2, les 32 bits du résultat faible sont rangés dans le registre GR2:

```
cycle                          opération
-------------------------------------------------------------------
t-1         chargement des opérandes dans l'ADD3 et préparation de
            la 1ère phase de la multiplication (au niveau du BSH):
            BUSA := (GR1);
            BUSB := (GR2)
            U1 := BUSA;
            U2 := 4*BUSA
            U3 := 0;
            BSH.G := BUSB;
            BSH.D := 0;
            opBSH  = SRH(4);
            PLA := BSH(28:4);  décalage-droite de 4 bits
            opADD3 = plus;

t+0         BUSB := SADD3; U3 := BUSB/16;
            BSH.D := BUSB; bA2 := bA1;
            BSH.G := SBSH;
            opBSH = SHR(4);
            PLA := BSH(28:4);  décalage droite de 4 bits
            opADD3 = plus;
                   :
                   :
                   :
                   :
                   :
t+8         BUSB := SADD3; U3 := BUSB/16;
            BSH.D := BUSB;
            BSH.G := SBSH;
            opBS = SHR(4);       à ce niveau on a Rés.faible=BSH.G
                                 et Rés.fort=U3

t+9         chargement des résultats et tests
            SBSH := BSH.G; BUSA := SBSH; (GR2) := BUSA;
```

IV.2. REPRESENTATION DES NOMBRES BINAIRES A VIRGULE FLOTTANTE

Les nombres flottants sont apparus lorsqu'il fallut représenter sous une forme à la fois concise et précise le large éventail des nombres réels du très grand au très petit. Dans la représentation adoptée, souvent appelée notation scientifique en raison de son domaine de prédilection, tout nombre flottant est composé d'un triplet: son signe, son exposant et sa mantisse. La mantisse est constituée des chiffres non nuls les plus significatifs. L'exposant, appelé aussi caractéristique, est un facteur multiplicatif indiquant la la position réelle de la virgule dans le nombre: d'où l'appellation de nombre à virgule flottante, ou de nombre flottant. Le bit de signe est celui qui indique le signe du nombre.

IV.2.1. LA BASE DE REPRESENTATION

La base de représentation d'un nombre flottant que nous adoptons est la représentation hexadécimale (format IBM 360/370) donnée par les spécifications du cahier des charge de notre processeur. Cette représentation est meilleure que la représentation binaire (DEC PDP-11, Cray-1, Standart IEEE, etc...) pour deux raisons: d'une part sa meilleure adéquation au format interne des machines, car la frontière caractéristique-mantisse se retrouve sans difficulté à la frontère d'octet, d'autre part sa plus grande vitesse dans les opérations de normalisation ou de dénormalisation: les décalages se faisant 4 bits par 4 bits, il ne faut, par exemple, pas plus de sept décalages pour normaliser/dénormaliser une mantisse de 56 bits (surtout que ces opérations sont très fréquentes), on verra par la suite comment on peut réduire encore plus ce nombre de décalage avec l'opérateur BSH (Barrel SHifter).

Un nombre en virgule flottante est donc représenté en machine par un

signe, un exposant et une mantisse dont la taille et la forme sont liées à l'architecture interne du processeur.

Format	Champs	Base	Bit implicite	Nombres négatifs
Dec PDP-11	1-8-55	2	oui	SVA
Cray-1	1-15-48	2	non	SVA
CDC 7600	1-11-48	2	non	C1
IBM 7030	1-11-48	2	non	SVA
IBM 360/370	1-7-56	16	non	SVA
Sel	1-7-56	16	non	C2
Standard IEEE	1-11-52	2	oui	SVA

(Réf : note d'application AN-111 par E. Gordon & C. Hastings, MMI).
Un champ de 1-8-55 signifie un bit de signe, huit bits de caractéristique, et cinquante-cinq bits de mantisse ; pour les nombres négatifs, SVA signifie Signe + Valeur absolue, C1 complément à un, et C2 complément à 2.

Tableau3: quelques formats flottant 64 bits

LA VALEUR

La valeur des nombres à virgule flottante est définie par l'equation:

$$V = (-1)^{S} \, 16^{E} \, M$$

$$E = C-64$$

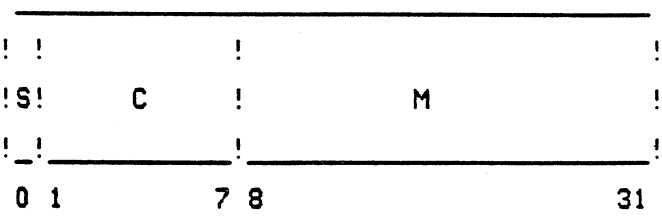
S est le signe du nombre flottant, E est son exposant, C est sa caractéristique et M est sa mantisse.

La valeur zéro est représentée par un nombre à virgule flottante dont la mantisse est égale à zéro.

LE FORMAT ADOPTE

Un nombre à virgule flottante court occupe 4 octets consécutifs. Il

est représenté dans le format suivant:



LE SIGNE

Il est contenu dans le bit zéro:

S = 0 signe positif

S = 1 signe négatif

LA CARACTERISTIQUE C, L'EXPOSANT E

La caractéristique C d'un nombre flottant représente un exposant signé, plus précisément la puissance entière à laquelle la base de représentation doit être élevée avant la multiplication de la mantisse. La technique retenue en général est celle de l'exposant biaisé: une valeur constante est ajoutée à l'exposant vrai signé; cette constante est telle que l'exposant vrai le plus négatif soit représenté par une caractéristique nulle. Cette méthode offre le gros avantage de transformer un entier signé (l'exposant initial) en un entier non signé (la caractéristique). Quand au biaisage proprement dit, il est simple à réaliser: il suffit de complémenter le bit de signe de l'exposant initial.

La question peut de poser, de savoir pourquoi la représentation flottante place la mantisse à droite de la caractéristique, alors que pour la notation scientifique, plus ancienne, elle est à gauche. L'explication est la suivante: le signe étant à gauche, la caractéristique (exposant biaisé) au milieu et la mantisse normalisée à droite, la comparaison de deux nombres flottants peut s'effectuer très rapidement par soustraction binaire directe, sans tenir compte des différents champs du triplet. Cela est dû au fait que l'ensemble des

nombres flottants ainsi constitué a le même ordonancement que celui des nombres binaires, fractionnaires ou entiers. Cette propriété justifie d'autant plus l'usage de l'exposant biaisé.

L'interêt de cette méthode immédiate de comparaison n'est pas à négliger: toute opération d'addition ou de soustraction de deux nombres flottants débute par une phase de comparaison des deux caractéristiques, le plus petit des deux nombres devant être dénormalisé jusqu'à l'égalisation des caractéristiques.

Il ne suffit pas de décrire les composantes d'un nombre flottant, il faut également les dimentionner. C'est en effet la taille de la caractéristique qui limite l'intervalle de représentation, et la taille de la mantisse celle de la précision.

La caractéristique est contenue dans les bits 1 à 7. Elle varie donc de 0 à 127. L'exposant E est une puissance de 16 qui varie de -64 à +63 (E = C-64).

LA MANTISSE

La mantisse est un nombre fractionnaire (sauf pour les séries 6600/7600 et Cyber de CDC, où la mantisse représente un nombre entier) dont la virgule est immédiatement à gauche du chiffre de plus fort poids, c'est-à-dire que la partie entière est nulle.

Pour un nombre flottant court, sa mantisse est contenue dans les bits 8 à 31, ce qui représente 6 digits (4 bits) hexadécimaux. Un nombre flottant est dit long si sa mantisse occupe 56 bits. La base de séparation flottante se trouve à la position la plus forte du digit le plus à gauche de M.

Emplacement dans la mémoire - Il est spécifié par l'adresse de l'octet le plus à gauche.

NORMALISATION

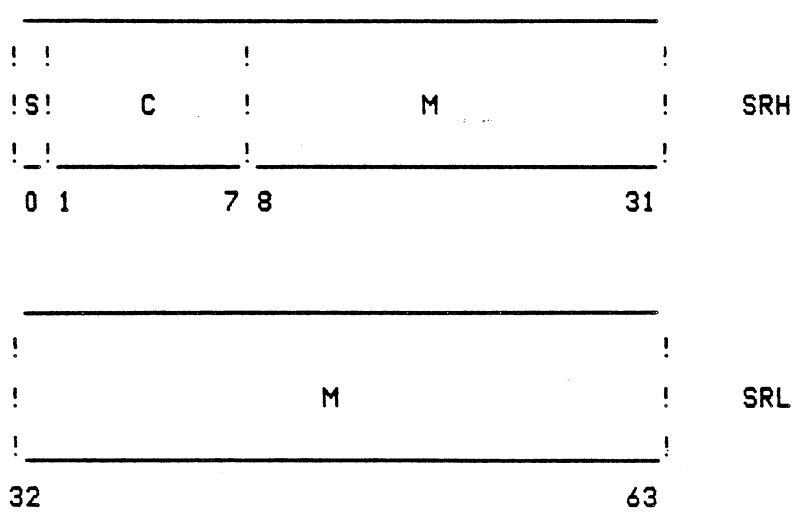
Pour disposer de la plus grande précision possible, la mantisse doit être normalisée. Un nombre binaire à virgule flottante est dit normalisé lorsque le digit hexadécimal le plus à gauche de sa mantisse est différent de zéro. Si la base de représentation est hexadécimale, le premier digit d'une mantisse normalisée est donc nécessairement égal au moins à "1".

La normalisation consiste à décaler à gauche la mantisse par groupe de 4 bits jusqu'à ce que le digit hexadécimal le plus à gauche soit différent de zéro. La caractéristique est réduite alors par le nombre de digits hexadécimaux décalés et des zéros remplaçent les bits vacants dans la position la plus à droite de la mantisse.

Remarque: tous les nombres flottants en mémoire sont préalablement normalisés.

IV.2.2. DEFINITION DES REGISTRES SCIENTIFIQUES SR

Ce sont 4 registres scientifiques SR de 64 bits qui sont éclatés en 8 registres de 32 bits dont 4SRH et 4SRL qui se présentent réspectivement sous les formats:



Un flottant court occupe le contenu de SRH (32 bits).
Un flottant long occupe le contenu de (SRH)U(SRL) (64 bits).
Un flottant étendu occupe deux registre SR (120 bits).

Les trois types de flottants peuvent se trouver en mémoire mais seuls les longs et les étendus peuvent être contenus dans les registres SR; d'où la conséquence : lorsqu'on a un flottant court qui occupe le registre SRH, le registre SRL est mis à zéro.

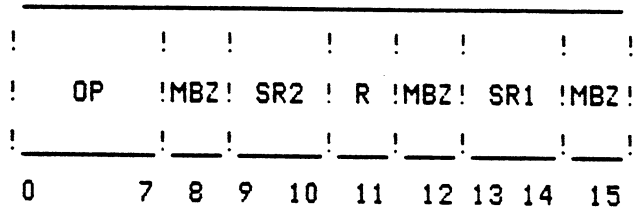
Les SR sont sélectionnés par les champs SR, SR1 ou SR2 de l'instruction. Les résultats intermédiaires sont sur 15 (ou 29) digits hexadécimaux selon que le flottant est long ou étendu. Le dernier digit (Guard digit) hexadécimal, qui représente le poids faible du registre, intervient selon la valeur de R (champs qu'on voit apparaître dans les différents instructions) pour arrondir ou tronquer ce résultat.

IV.3. ANALYSE DES INSTRUCTIONS

Il sont de plusieurs familles distinctes:

1. Instructions registre à registre: SRSR, GRGR, GRSR, etc...

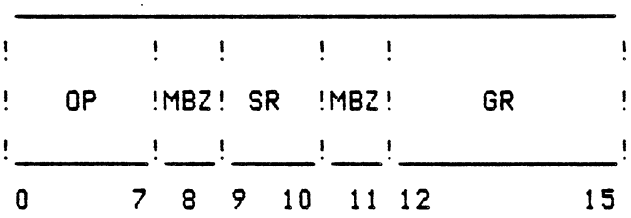
Toutes ces instructions s'exécutent en 2 cycles visibles t+0 et t+1. Les instructions de la famille SRSR par exemple ont le format suivant:



Ces instructions s'exécutent, en principe, sur des opérandes lus dans les registres scientifiques SR, le résultat étant chargé dans le registre SR2. Le champ R indiquera si on a besoin d'arrondir ou de tronquer le résultat et le champ MBZ est nul.

-Instructions du groupe SRGR

Elles se présentent sous le format:



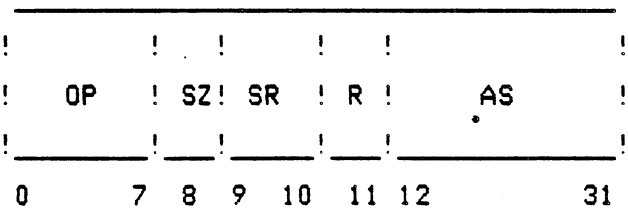
Ce sont des instructions qui s'exécutent entre des registres SR et des registres GR. Deux instructions sont concernées:

L'instruction LISG qui permet de ranger le nombre à virgule fixe, contenu dans le registre GR, dans le registre SR après l'avoir transformé en un nombre flottant.

L'instruction STISG par contre range le contenu d'un registre SR dans le registre GR après transformation du nombre flottant en un nombre à virgule fixe.

2. Instructions registre à mémoire ou vis-versa comme SRX

Elles ont le format suivant:



OP est le champ code opération
AS est le champ adresse syllabe

Ces instructions s'exécutent,en principe, sur un opérande lu en mémoire et un registre SR. Le résultat est rangé dans le registre SR. Toutefois, il faut noter des exceptions à ce shéma général: des familles d'instructions ne lisent pas d'opérandes en mémoire mais rangent le contenu d'un registre ou d'un résultat à l'adresse calculée.

Afin de faciliter l'implantation et l'accélération du calcul flottant au niveau de la partie opérative, il nous parait intéressant d'éclater les registres SR 64 bits en un banc de registres de 32 bits où 4 registres contiendront signe, caractéristique et mantisse poids fort et les 4 autres registres vont contenir les 32 bits de la mantisse poids faible. Cela nous permettera, dans le cas d'un décalage ou d'une normalisation par exemple, de charger en un seul cycle les deux mots de 32 bits à la fois au niveau de l'opérateur concerné.

Dans le but d'accélération du calcul, nous proposons d'utiliser un décaleur ("barrel shifter" BSH) à 2 tampons d'entrés BSH.G, BSH.D. Ceci va nous permettre, dans le cas d'une addition par exemple, de charger en un seul cycle le contenu de deux registre SR à la fois, à savoir,les 2 matisses poids faible et les deux mantisses poids fort pour qu'au cycle suivant on sélectionne la mantisse la plus faible pour un décalage.

cycle 1 — chargement des caractéristiques pour comparaison;
chargement des mantisses poids faible sur BSH.G;
chargement des mantisses poids fort sur BSH.D;

cycle 2 — si C1 est différent de C2 alors on sélectionne la mantisse correspondant à un Ci petit pour décalage à droite et génération d'une constante à ajouter à cette caractéristique Ci.

En ce qui concerne la normalisation, il suffit d'avoir un encodeur de zéros (SAM) qui est un réseau de portes analysant la mantisse 4 bits par 4 et qui va commander le nombre de décalage à faire du résultat qui se trouve à l'entrée du BSH.

CHAPITRE V

RESSOURCES MATERIELLES DE BASE

V - RESSOURCES MATERIELLES DE BASE

Après avoir établi le schéma de quelques blocs fonctionnels de la Partie Opérative, on va définir les schémas logiques et électriques de base à partir desquels on construit les blocs de la P.O (registres, opérateurs etc...). De là on établira la stratégie d'implantation de la circuiterie CMOS (ou "complementary MOS" en anglais).

La validation des différents blocs fonctionnels doit faire l'objet de simulations électriques très poussées, garantissant ainsi les options choisies.

V.1. LES PORTES DE BASE UTILISEES

Le but de ce paragraphe est de rappeler brièvement les éléments de base nécessaires à la conception des circuits logiques intégrés en technologie CMOS.

V.1.1. LES TRANSISTORS

Le principe de la technologie CMOS est de combiner un transistor MOS canal N avec un transistor MOS canal P, implantés simultanément sur un même substrat. Ces transistors fonctionnent tous les deux comme des transistors signaux à enrichissement. L'asymétrie des temps de propagation est dûe à la différence des mobilités des porteurs de charges P et N et aux rapports des géométries des transistors (MEA). Nous représentons dans la figure V.1 le principe de base de la technologie CMOS.

ouerdani:Tue May 13 19:14:24 1986
figV1

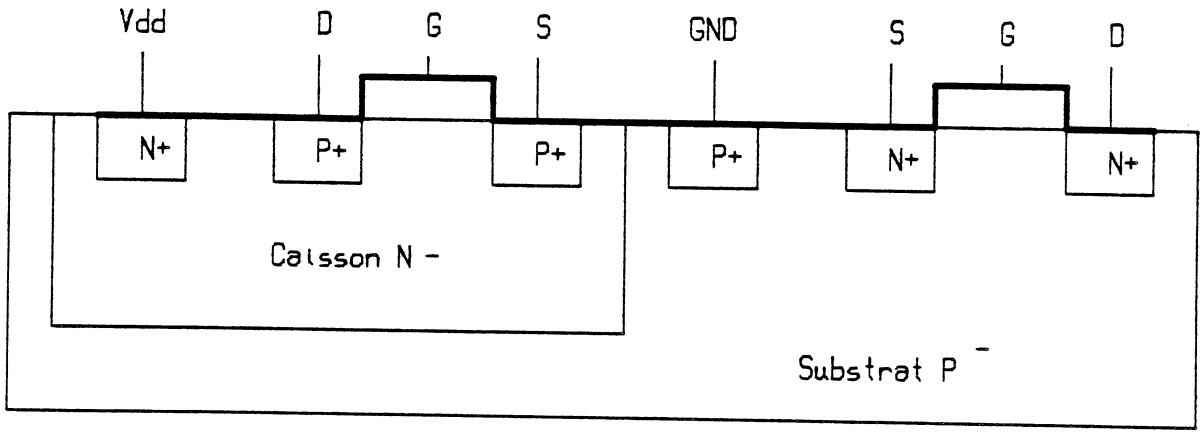


Figure V.1: principe de la technologie CMOS à caisson N

Cette technologie est plus complexe que la technologie NMOS E/D (Enrichie/Deplété) au niveau des opérations technologiques. En effet, il s'agit de combiner deux transistors de types différents à partir d'un même substrat, c'est-à-dire de créer artificiellement un substrat N sur un substrat P ou inversement, ce qui amène les technologues à utiliser un caisson dont il faut respecter les limites lors des implantations, d'où la difficulté pour la réduction et l'optimisation des surfaces à implanter. Néanmoins, cette technologie présente l'avantage appréciable d'une faible consommation électrique puisque la consommation statique est nulle et que seule une consommation dynamique subsiste.

Ces deux types de transistors sont à enrichissement. Ils diffèrent par le fait que le type N conduit sur le niveau haut de sa tension de grille (G=5 volts) alors le type P conduit sur le niveau bas (G=0 volt); On dit que ces deux transistors sont complémentaires.

Notation : afin de différencier aisément les deux transistors, on adoptera les notations symbolisées de la façon suivante (cf figure V.2):

ouerdani:Tue May 13 19:14:56 1986
figV2

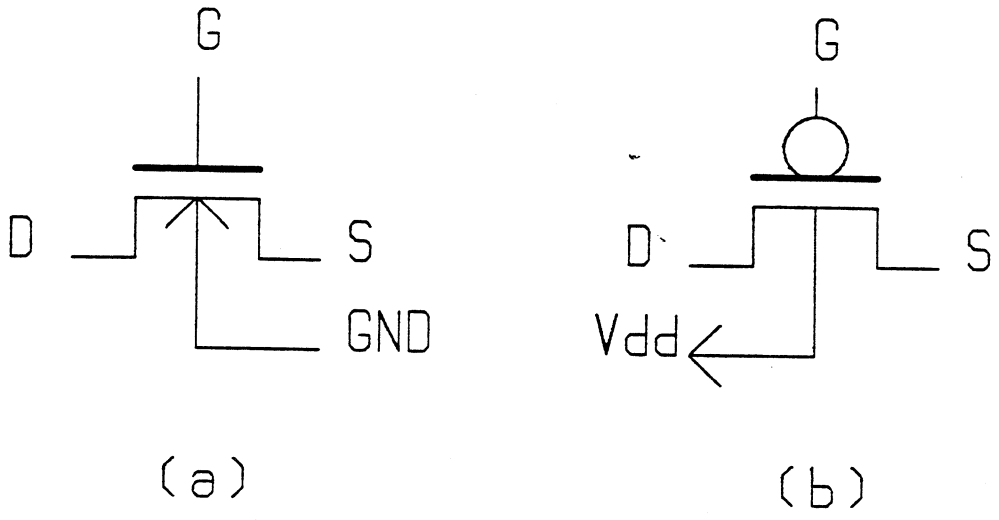


figure V.2: Représentation symbolique des transistors
(a) transistor N (b) transistor P

V.1.2 PORTES DE PASSAGE

Cet opérateur est très important en circuiterie MOS puisque la réalisation de fonctions logiques complexes peut être parfois réduite à un réseau de portes de transmission qui contrôlent le transfert des données. On distingue deux sortes d'interrupteurs:

- <u>porte de passage simple</u>

C'est l'utilisation du transistor N en tant qu'interrupteur comme en technologie NMOS avec comme caractéristique principale la dégradation possible d'une tension de seuil en sortie et une disymétrie dans les temps de propagation des fronts descendants et montants. Il est souvent représenté par le symbole suivant (cf figure V.3).

ouerdani:Tue May 13 19:15:22 1986
figV3

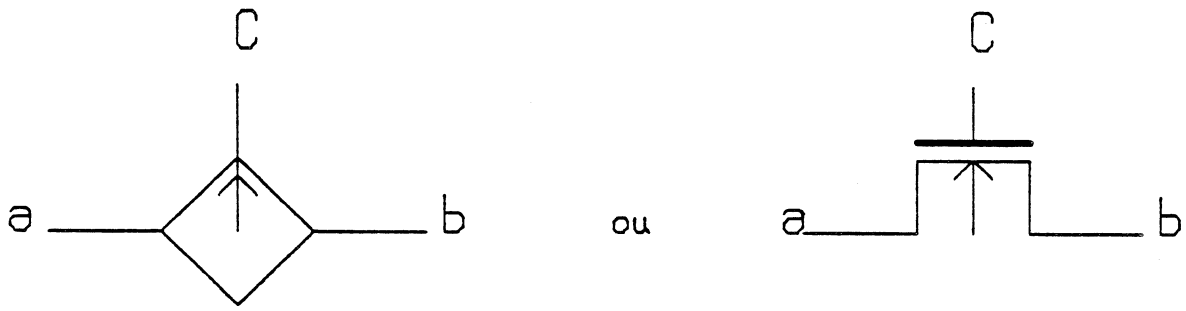


Figure V.3: porte de transmission NMOS

Le transfert de charges se fait d'une façon bidirectionnelle. Un nivau haut de la commande C autorise le transfert de l'information de a vers b ou vice versa. Dans le cas contraire (C=0), l'interrupteur est ouvert; les noeuds a et b sont isolés électriquement entre eux.

- <u>porte de transmission complémentaire</u>

C'est un interrupteur CMOS composé de deux transistors complémentaires montés en parallèle avec sources et drains communs. Ce type d'interrupteur, bien que nécessitant deux lignes de sélection S et S* (donc coûte cher) restitue les signaux en sortie dans leur intégralité (pas de perte de seuil). Nous le présentons de la façon suivante (cf figure V.4):

ouerdani:Tue May 13 17:48:08 1986
figV4

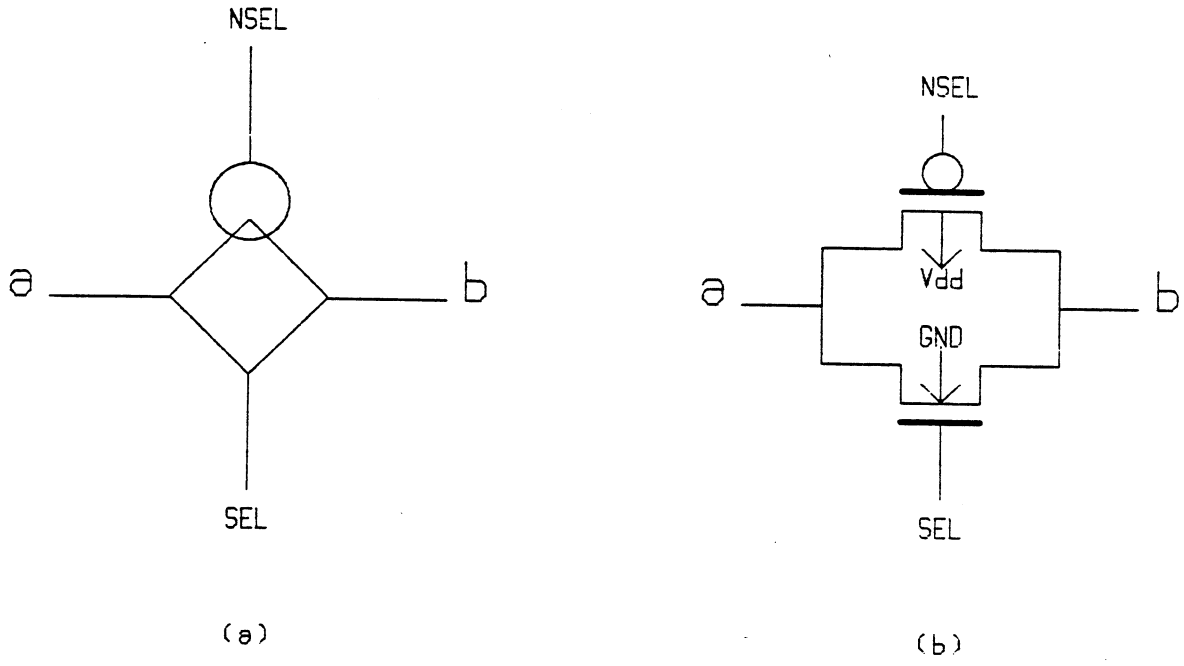


figure V.4: Représentation d'un interrupteur CMOS
(a) symbolique (b) électrique

On remarque que lorsque le signal SEL est à l'état haut les deux MOS sont conducteurs, la sortie a reçoit l'information b à travers ces deux transistors; par contre lorsque SEL est à l'état bas, les deux transistors étant bloqués, les noeuds a et b sont isolés, la porte est donc bloquée.

V.1.3. L'INVERSEUR CMOS

L'inverseur CMOS est constitué de deux transistors de base montés en parallèle. Le substrat du MOS P est connecté au potentiel le plus haut, soit Vdd, et celui du MOS N au potentiel le plus bas, soit la masse.

Les grilles des deux transistors sont commandées en parallèle par le même signal de commande In. La sortie Out se trouve au point commun des drains.

ouerdani:Tue May 13 19:26:48 1986
figV5

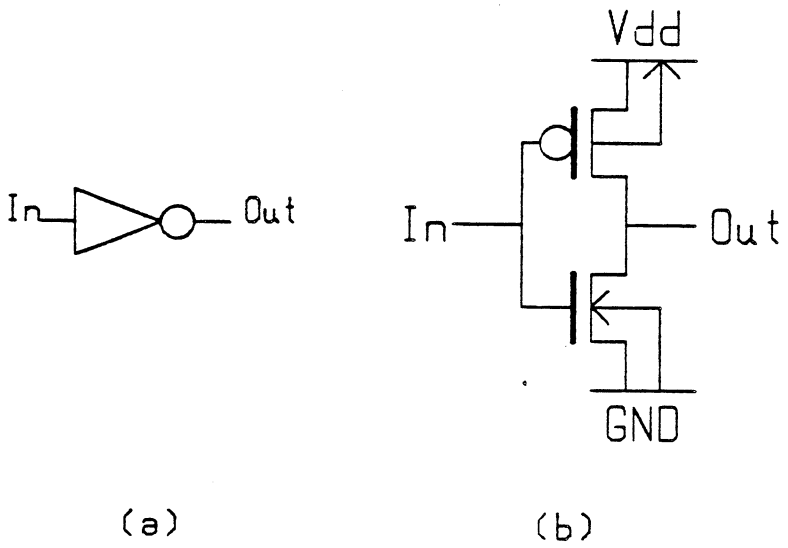


figure V.5: Représentation d'un inverseur en technologie CMOS
(a) symbolique (b) électrique

On remarque que l'inverseur CMOS est un montage où l'élément de charge est actif et non plus passif comme en technologie NMOS E/D

V.1.4. LA PORTE NON-ET

La mise en série de "n" transistors NMOS et la mise en parallèle de "n" transistors PMOS permet de réaliser une porte NON-ET CMOS à "n" entrées. En effet, la sortie est au niveau logique "0" si et seulement si toutes les "n" entrées sont au niveau logique "1"; c'est-à-dire que tous les transistors N sont conducteurs et tous les transistors P sont bloqués. Une porte NON-ET CMOS à deux entrées est présentée en figure V.6.

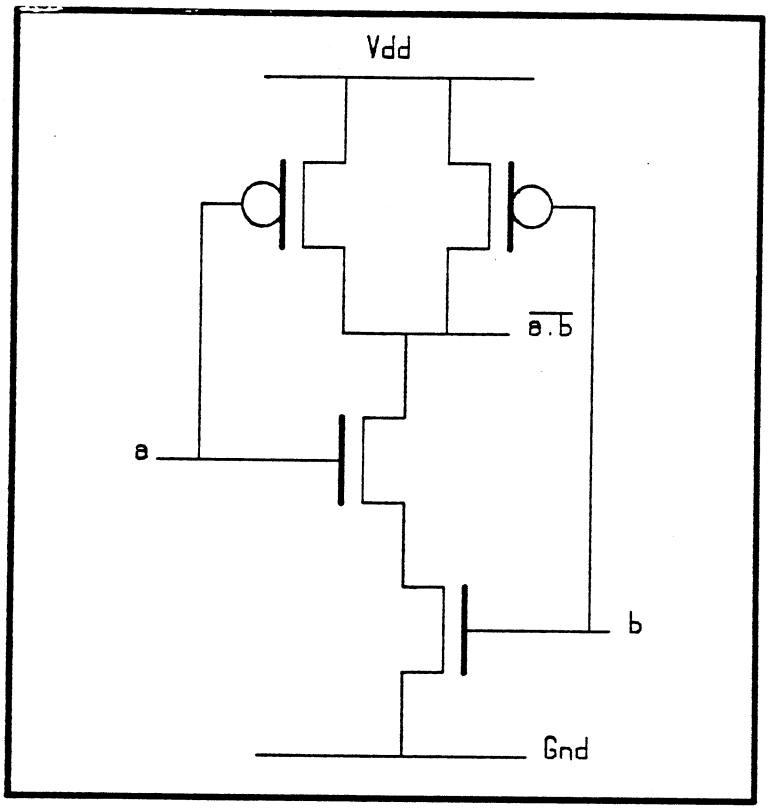
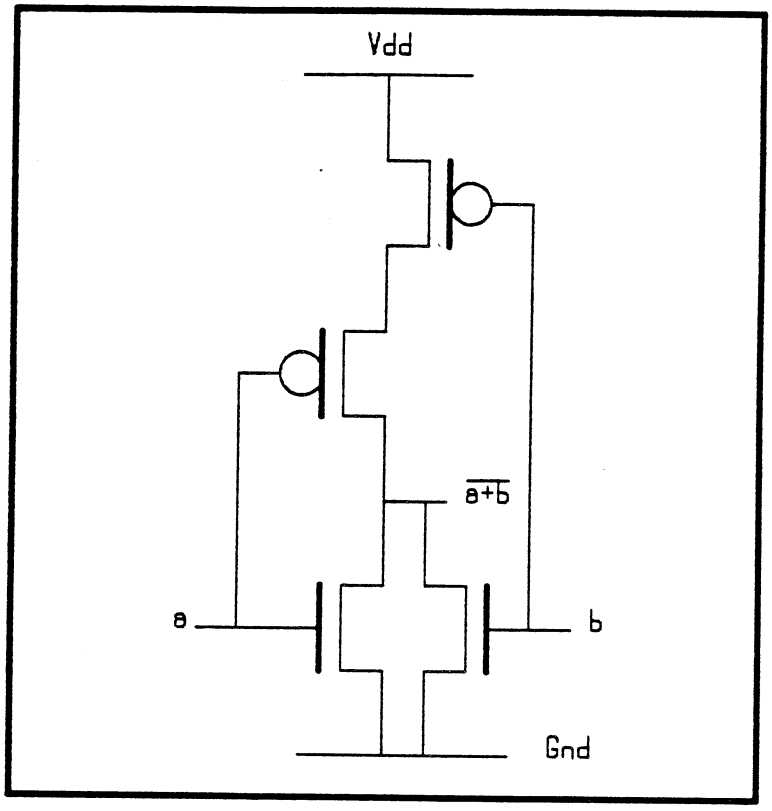


Figure V.6: porte NON-ET CMOS à 2 entrées

V.1.5. LA PORTE NON-OU

La mise en parallèle de "m" transistors NMOS et la mise en série de "m" transistors PMOS permet de réaliser une porte NON-OU à "m"entrées. En effet, la sortie est au niveau logique "1" si et seulement si toutes les "m" entrées sont au niveau logique "0"; c'est-à-dire que tous les transistors P sont conducteurs et tous les transistor N sont bloqués. Une porte NON-OU CMOS à 2 entrées est présentée en figure V.7.



ouerdan1:Tue May 13 19:27:14 1986
figV7

Figure V.7: porte NON-OU CMOS à 2 entrées

V.1.6. LE OU-EXCLUSIF

Cette porte peut être réalisée à l'aide de l'inverseur CMOS et de la porte de transmission simple ou complémentaire. Mais il existe une autre solution avec une seule porte de transfert qui est aussi intéressante (cf figure V.8):

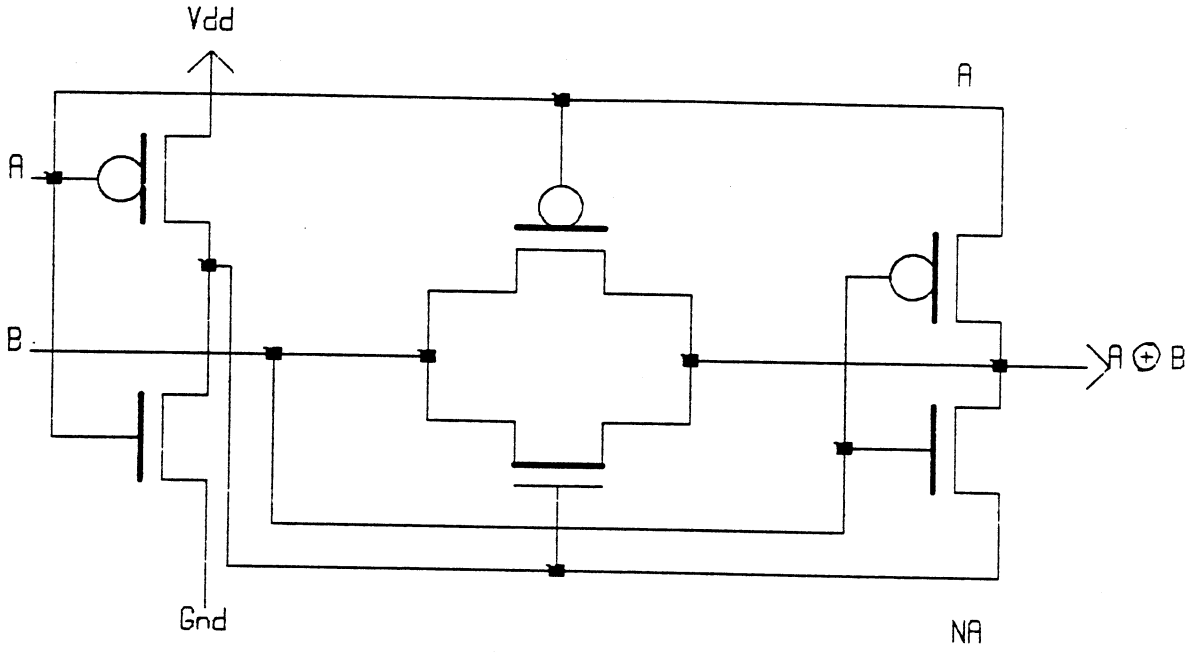


Figure V.8: Porte OU-exclusif en technologie CMOS

Cette fonction est obtenue en utilisant une porte de transmission complémentaire, un inverseur CMOS usuel et un autre inverseur CMOS modifié car son alimentation est fournie par les entrées A et NA.

V.2. LES OPERATEURS STATIQUES

Il permettent de synthétiser une fonction logique sous forme de produits et de sommes selon le même principe de base que les opérateurs NMOS E/D. Ils sont constitués d'un réseau de transistors NMOS enrichis et d'un réseau de transistors PMOS dual du réseau N qui assure la complémentation (cf figure V.9).

ouerdani:Tue May 13 17:5Ø:59 1986
figV9

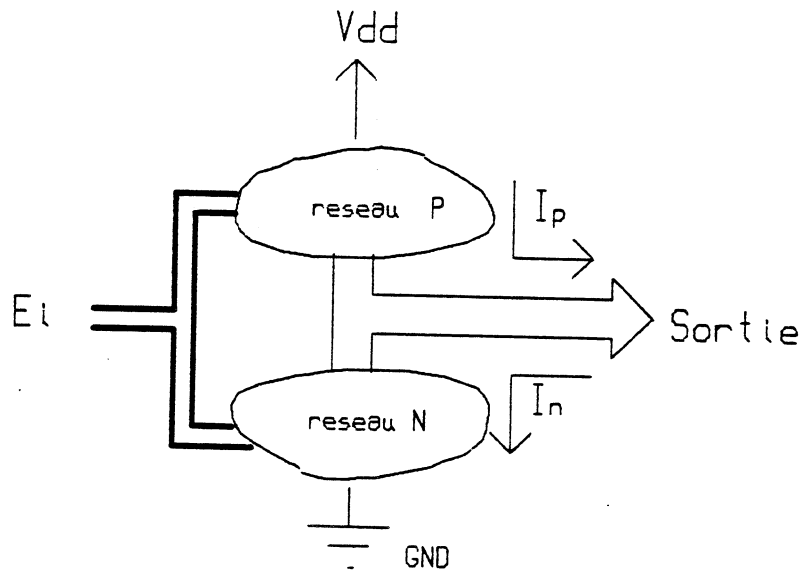


Figure V.9: principe d'implantation des portes logiques CMOS

Le réseau P dual du réseau N permet de conserver les principales caractéristiques de consommation statique nulle de l'inverseur CMOS (cf figure V.10).

ouerdani:Tue May 13 17:51:26 1986
figV1Ø

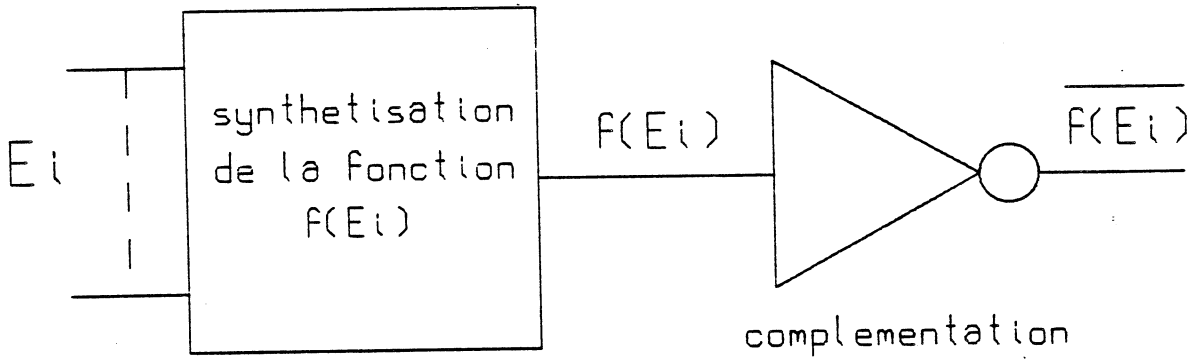


Figure V.10: autre approche de la figure 9

Comme en technologie NMOS, ces opérateurs sont une extension de

l'inverseur statique CMOS; le réseau de transistors P délivre un courant pour charger la capacité de sortie pendant que les transistors de réseau N sont bloqués et pour la décharge de cette capacité, le réseau N ouvre une voie de passage pour ce courant vers la masse pendant que les transistors du réseau P sont bloqués.

Chaque entrée commande à la fois un MOS N et un MOS P mais ce signal de commande commun rend soit le MOS P conducteur et le MOS N bloqué soit l'inverse suivant s'il est à l'état bas ou à l'état haut. A chaque somme de produits du réseau N il faudra donc faire correspondre un produit de sommes du réseau P et réciproquement.

V.3. PRINCIPES TOPOLOGIQUES

La structure la plus répandue dans la plupart des microprocesseurs existant actuellement sur le marché pour l'implantation d'une Partie Opérative consiste à dessiner une nappe de fils rectilignes et horizontaux en aluminium, matériaux le moins résistif de tous les conducteurs (véhiculant plus facilement l'information d'un bout à l'autre du chemin des données). Sous cette nappe de fils on dessine toute la logique de la Partie opérative. Cette logique communique avec cette nappe de fils par l'intermédiaire de contacts appropriés pour réaliser des interconnexions entre les différents éléments actifs du circuit. Cette technique classique nous semble une bonne stratégie que nous allons adopter, avec une technologie à deux niveaux de métalisation qu'on appelle Métal1 et Métal2, ce dernier étant celui du niveau supérieur (ou encore le plus haut niveau sur le silicium).

V.3.1. LE BIT SLICE

Le principe architectural qui prédomine pour l'implantation de la Partie Opérative d'un processeur est celui du découpage en tranches de un Bit (Bit Slice), de hauteur fixe, implantées sous une nappe fixe de fils en Métal2 qui portent les différentes alimentations et les bus de

données, tandis que les commandes venant de la partie contrôle seront disposées verticalement en Métal1 et vont piloter les grilles des transistors commandés des différents éléments de la Partie Opérative (registres, opérateurs, ...). Nous représentons cette disposition suivant la figure V.11:

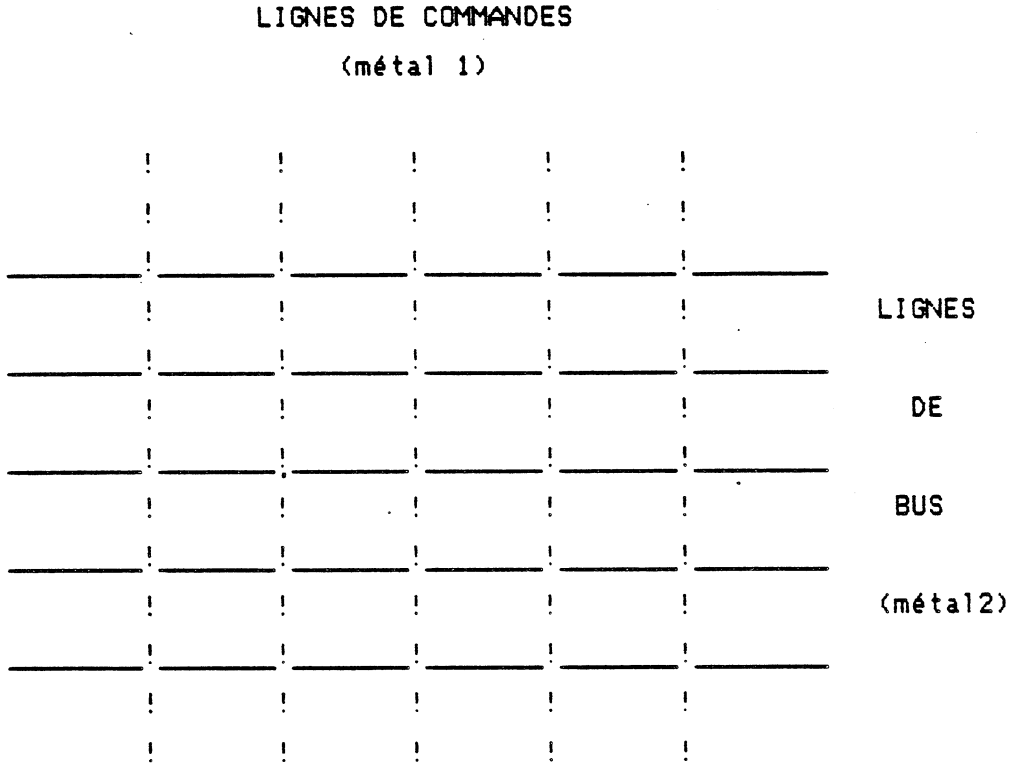


Figure V.11: disposition des niveaux de métalisation

Grâçe à cette topologie, la Partie Opérative du dit processeur peut être composée de plusieures parties opératives élémentaires, pouvant être accolées et commandées par la même partie contrôle, puisqu'elles ont la même topologie; chacune assurera une fonction: une constante, une variable mémorisée, etc... et opérant sur un seul des bits du mot, ce qui nous donne un agencement tranche par tranche, le nombre de ces tranches étant égal à 32 puisque les données manipulées sont de 32 bits.

L'intérêt de cette architecture apparaît lors de l'assemblage final du circuit. On dessine une seule tranche de Partie Opérative qui va constituer la cellule de base et le dessin final s'obtient par des répétitions de la première tranche. Ces répétitions de dessin sont possibles avec tous les systèmes de dessin simples (éditeurs graphiques) ou évolués comme LUBRICK (réf. SCH-85). Ceci amène un gain de temps appréciable lors de la conception finale du circuit.

Dans cette topologie, il reste à définir une hauteur fixe (et par là un nombre déterminé de fils) du Bit Slice permettant le dessin de toutes les cellules de base à partir desquelles on fait l'assemblage des différents élément de la P.O. Le choix de ce paramètre est déterminant pour l'optimisation en surface de la P.O et influe sur ses performances. En fait ce choix est un compromis entre ces deux paramètres, à savoir l'optimisation de la surface (et restriction de l'architecture) et la performance de la machine.

Compte tenu des règles de dessin imposées par le fondeur du silicium -définissant une largeur minimale des lignes d'aluminium et l'écartement entre chaque ligne- et l'étude de l'implantation des différents points mémoire de la P.O, on est amené à choisir une solution à 9 fils par Bit qui peuvent supporter jusqu'à trois bus différentiels (bifilaires) et un bus monofils servant à connecter les divers registres et les opérateurs. Le bus monofil, dit aussi ligne de service, a été ajouté pour permettre la réalisation des ponts à l'intérieur d'une cellule et la connexion entre les différentes briques.

La hauteur de la brique est choisie selon des considérations électriques et technologiques et on notera que les lignes Vdd sont partagées entre toutes les briques voisines. Les fils portants les alimentations (Vdd et Vss) sont dimentionnés de manière à éviter les "rappels" au milieu de la P.O. Finalement, la hauteur d'une tranche et la nappe de fils fixe sont schématisées par la figure V.12 suivante:

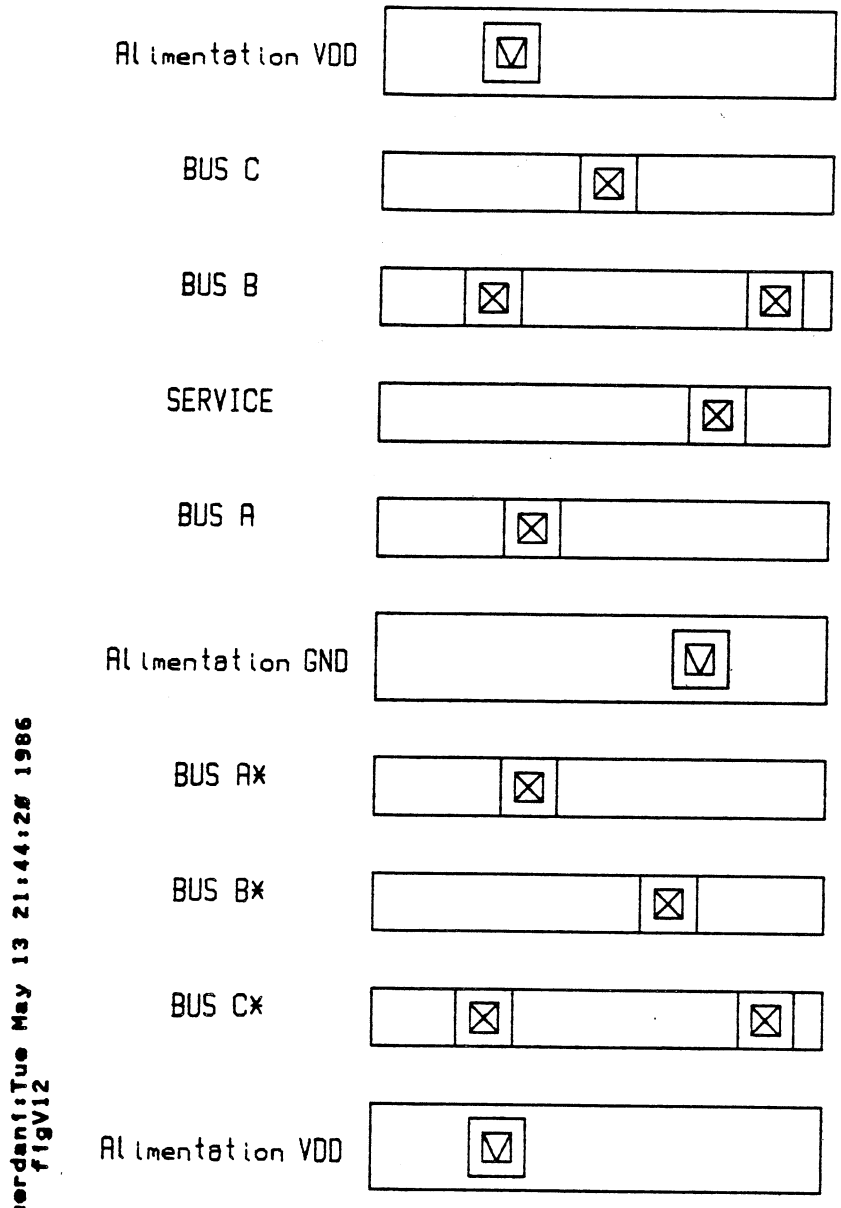


figure V.12: structure du Bit Slice

On remarquera que dans les microprocesseurs, les fonctions de mémorisation représentent une surface appréciable du circuit ce qui nous a amené a en tenir compte pour le choix de la hauteur du Bit Slice et le choix des Bus.

La technologie utilisée pour ce circuit ne permet pas les contacts entérés, comme dans certaines technologies NMOS, mais uniquement les contacts simples. ainsi, un contact diffusion-polyslicium sera réalisé par deux contacts simples diffusion-aluminium et polysilicium-aluminium

reliés entre eux par un pont métallique. Ce qui veut dire que seule la couche métal 1 est connectable à toutes les autres.

V.3.2. LES BUS

Ce sont les bus qui assurent toute la circulation de l'information à l'intérieur d'une Partie Opérative. De ce fait ils interviennent pour une large part dans la vitesse et donc dans la performance du produit.

Les bus relient entre eux différents registres, opérateurs et plots. Par conséquent, ils sont souvent très longs et présentent donc une charge capacitive très importante et donc une constante de temps RC élevée. Pour réduire cette constante, on les réalise en aluminium, métal très peu résistif par rapport au Si Poly.

Le plus souvent, on a cherché à effectuer plusieurs traitements à la fois ce qui engendre une performance plus grande grace à ce parallélisme. Pour ce faire, la solution consiste à subdiviser les bus en plusieurs tronçons reliés entre eux par des transistors de passage. Ces tronçons de bus sont différentiels, bifilaires et à précharge:

- les deux fils du bus sont portés à une tension de précharge préalablement étudiée (voir chapitre VI);

- chaque registre ou opérateur est connecté aux deux fils du bus, étant ainsi autorisé à décharger l'un de ces fils par l'intermédiaire d'une porte de passage simple de type N: on assiste ainsi à une décharge lente de l'un des fils;

- un amplificateur différentiel de type N est alors activé, dont le rôle est d'accélérer la décharge du fil qui a commencé à se décharger;

- un amplificateur différentiel de type P est activé quelque nanosecondes plus tard que le type N pour maintenir au niveau haut le

fils complémenté;

- lorsque la décharge est terminée, les deux fils du bus sont dans l'un des états logiques (0,1) ou (1,0), et un registre "destination" est sélectionné pour recevoir cette valeur.

V.3.3. MODELE FONCTIONNEL

Le modèle électrique du bus différentiel à précharge cité ci-dessus permet d'implanter le modèle fonctionnel suivant: chaque cycle-bus permet de réaliser un transfert d'une source (registre, sortie d'un opérateur, ...) vers une destination (registre, entrée d'un opérateur, etc ...).

V.3.4. MODELE TEMPOREL

La période du cycle de base de la machine est fixé pour atteindre une certaine performance et, en se basant sur les pires cas de charge de bus, ce cycle de base a été décomposé en quatre phases élémentaires identiques marquées par quatre horloges H0, H1, H2 et H3 (cf fig V.13).

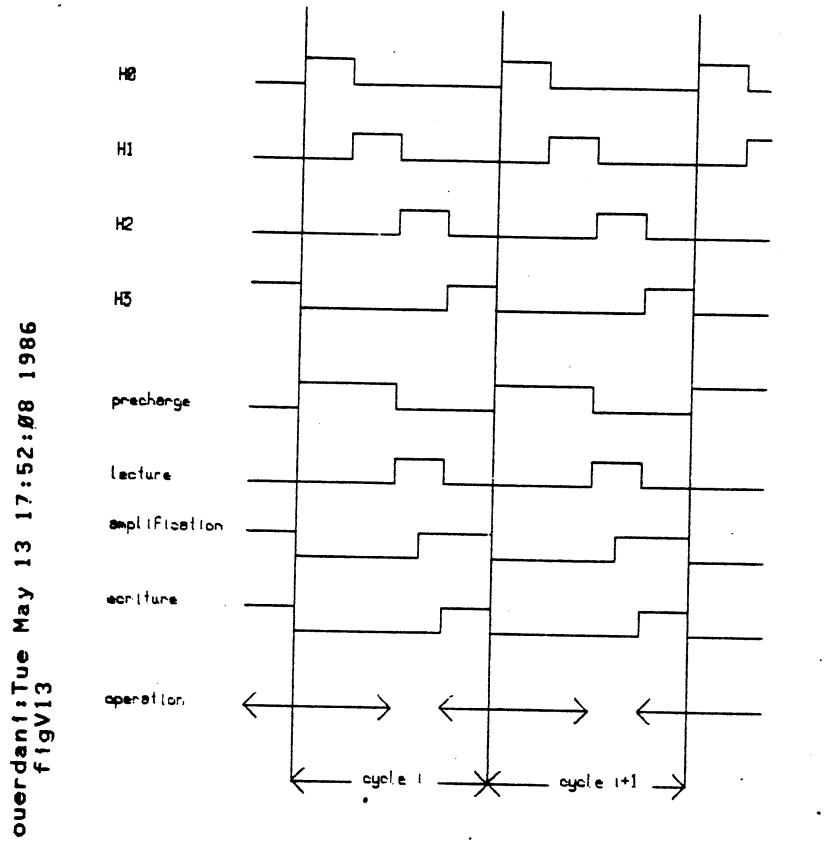


figure V.13: chronogramme de temps de la P.O

Le transfert d'informations d'un point de mémorisation vers un autre s'effectue selon la séquence suivante:

Phase HO et H1

Précharge des bus à une tension voisinant les 5 volts (on charge les capacités des deux fils du bus). Ces deux phases sont considérées comme des phases d'initialisation des bus à l'état logique "1".

Phase H2

Phase de lecture d'un point mémoire sur le bus, c'est-à-dire décharge de la capacité du bus vers le niveau logique "0" du point mémoire et enclenchement des différents amplificateurs différentiels (N puis P).

Phase H3

Phase d'écriture du bus dans un point mémoire. Ici, les capacités des deux bus "imposent" leurs valeurs à celles du point mémoire lequel est sélectionné en écriture. Le principe de ce transfert est représenté par la figure suivante:

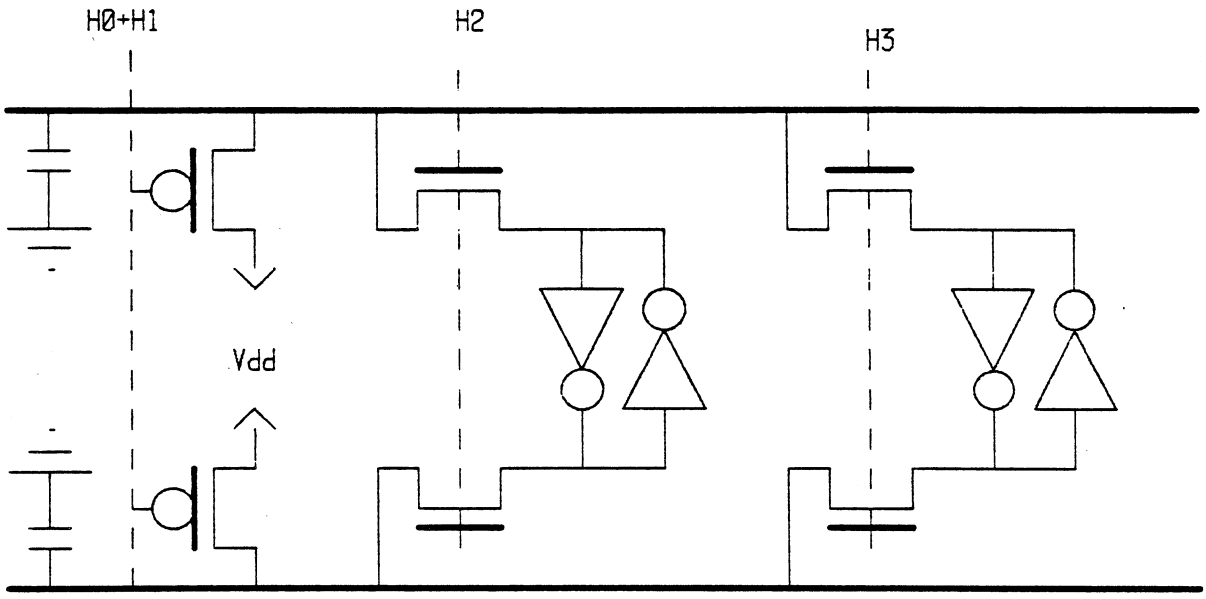


Figure V.14: opérations de transfert entre registre

On peut remarquer que le bus est assimilé à une mémoire dynamique puisqu'il mémorise temporairement l'information durant la phase H2 avant le transfert dans le point mémoire sélectionné pour l'écriture.

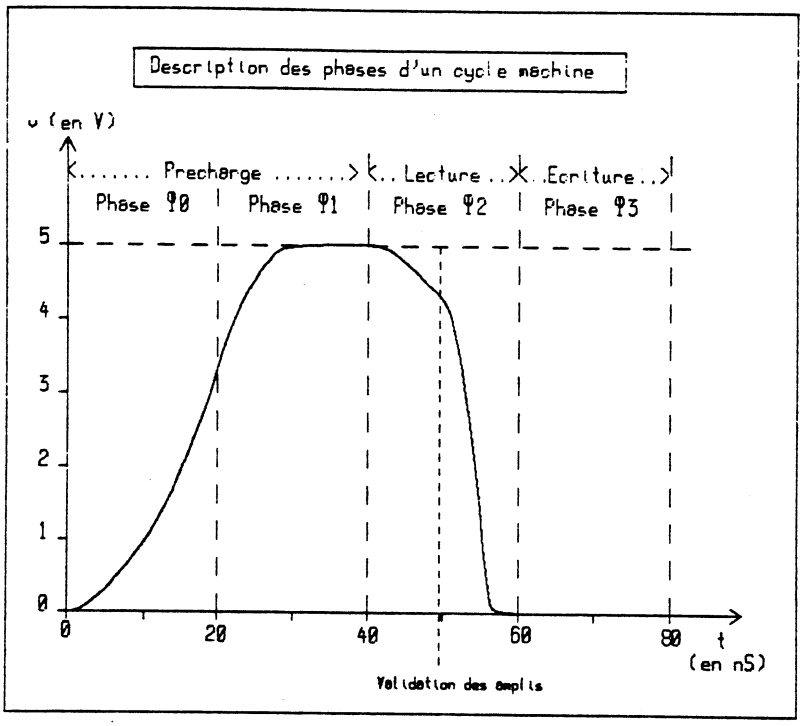


Figure V.15: description des phases d'un cycle machine

V.4. LES CELLULES DE BASE

V.4.1. PRESENTATION DU POINT MEMOIRE

La fonction mémoire a une importance considérable dans les circuits intégrés, soit par son aspect "circuit mémoire" à part entière en tant que composant "discret" (circuit le plus produit et le plus vendu), soit par son aspect "structure régulière" dans un circuit où elle peut apparaître sous de multiples formes.

Un point mémoire est un organe qui stocke une information pendant un certain temps, l'information étant accessible à travers un réseau d'interrupteurs à une certaine phase de temps du cycle machine. Cette information en mémoire peut être accessible par un ordre de lecture, ou modifiée par un ordre d'écriture.

Selon la durée de rétention de l'information dans le temps on distingue un mode de fonctionnement:

STATIQUE : l'information est stockée de façon permanente sans aucune

autre condition que le maintien de l'alimentation.

DYNAMIQUE : la rétention de l'information est assurée par la conservation des charges électriques en des points temporairement isolés du reste du circuit. L'existence des inévitables courants de fuite nécessite le recours à un rafraîchissement périodique de l'information.

D'une manière générale, on essaye d'organiser un circuit de la façon la plus régulière possible car celà entraîne:

--> Une plus grande facilité et une plus grande sécurité durant toutes les étapes de la conception.

--> La possibilité de modification du circuit. En effet il est plus aisé de changer une connexion dans une cellule de base que de reprendre l'implantation d'un bloc de logique complexe sans régularité.

--> L'accroissement de la densité lié au matriçage des structures très optimisées.

La structure des bus différentiels que nous venons de voir dans les principes topologiques permet facilement l'implantation des points de mémorisation de l'information.

L'emploi fréquent de point de mémorisation dans les circuits intégrés nécessite donc une structure simple, performante et peu encombrante.

Le point mémoire CMOS est constitué de deux inverseurs rebouclés (on dit aussi tête bêche) et connectables aux bus différentels par deux interupteurs comme le montre la figure V.16. Ces points de mémorisation sont auto-amplificateurs en ce sens qu'ils sont capables de décharger les bus auquels ils sont connectés

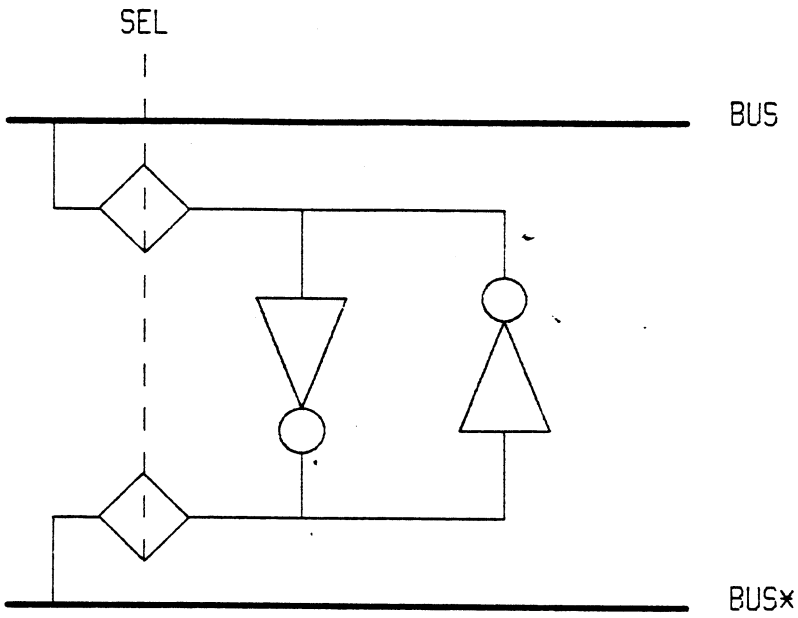


Figure V.16: point mémoire simple accès

On remarque que ce point mémoire est tout à fait symétrique et les commandes de sélection venant de la Partie contrôle peuvent être utilisées aussi bien pour la lecture à partir d'un bus que pour l'écriture vers un bus; ceci facilite l'implantation de ces points mémoires sur la structure en bus bifilaires. On peut noter également que le noyau de cet élément de mémorisation peut être en relation avec plus d'un bus mais seule une commande est utilisée à la fois. Le cas le plus fréquent qui se présente dans une Partie Opérative est le point mémoire à double accès de bus tel qu'on voit sur la figure V.17 suivante:

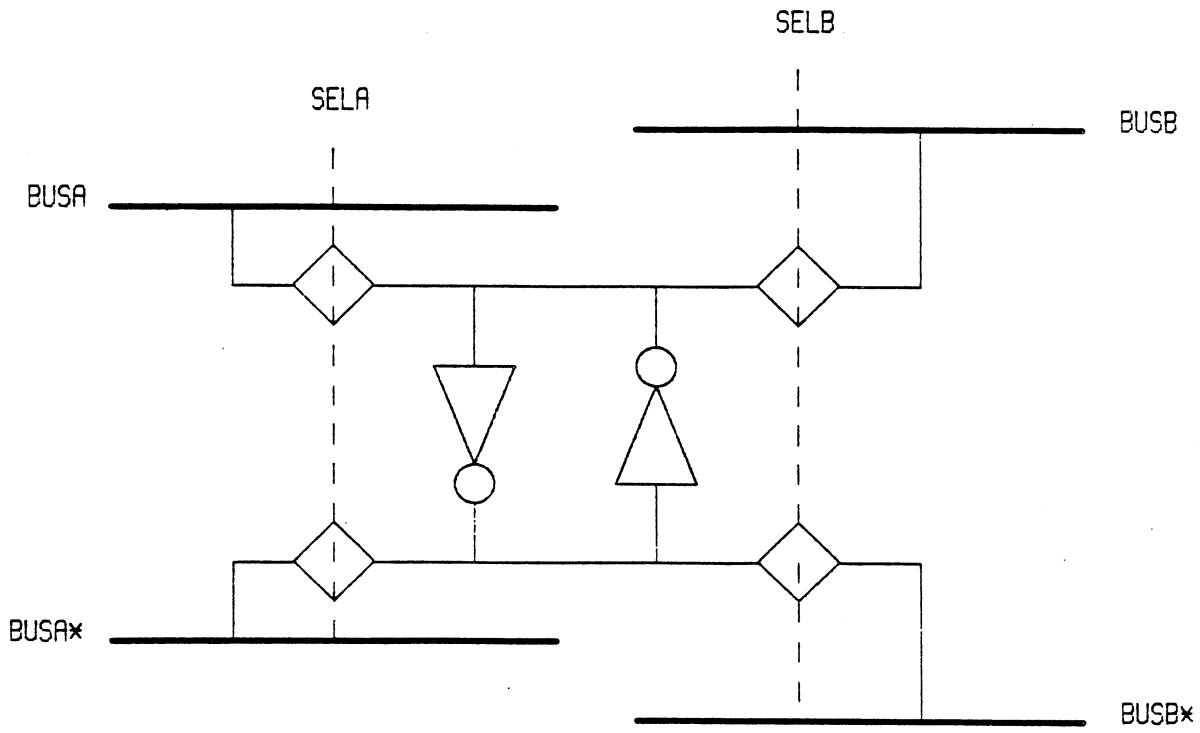


Figure V.17: point mémoire double accès

A partir de ce schéma simple on peut imaginer toutes les combinaisons possibles de connexion avec les bus. La lecture ou l'écriture dans ce point mémoire se fait par l'intermédiaire de portes de transmission pilotées par des lignes de commandes.

En circuiterie CMOS, on utilise les interrupteurs nécessitant deux lignes de sélection complémentées par porte (SEL et SEL*). Son avantage réside dans le fait qu'il a un comportement symétrique lors de la lecture d'un "1" ou d'un "0", mais dans notre cas, on peut utiliser un seul transistor de type N, puisqu'on a préchargé les bus.

Finalemant, le schéma de base d'un point mémoire à partir duquel on peut construire un registre double accès est représenté par la figure V.18:

ouerdani:Tue May 13 17:54:39 1986
figV18

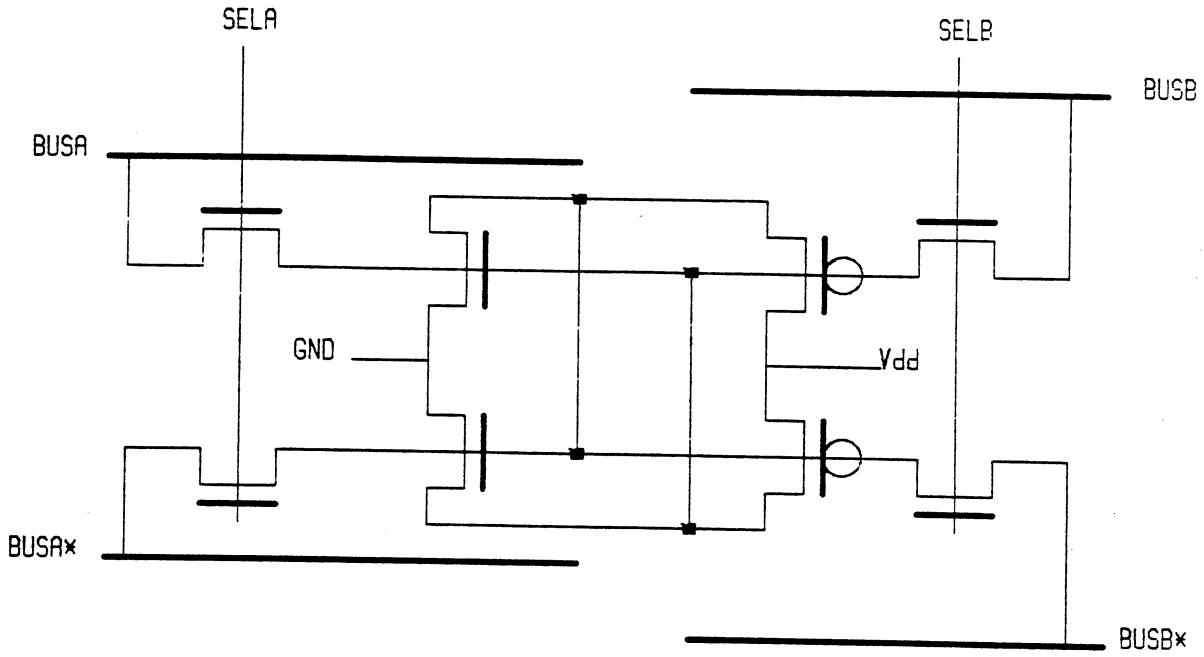


Figure V.18: représentation élerctrique du point mémoire

Les lignes de sélection SELA et SELB sont pilotées par les phases de lecture ou d'écriture.

V.5. PRESENTATION DE L'OPERATEUR DE DECALAGE

Le décaleur BSH ou "Barrel SHifter", permet d'effectuer n décalages (0<=n=<31) à gauche et à droite ainsi que des rotations. Il s'agit d'un circuit matriciel de commutation ("cross bar") dont chaque point de croisement est réalisé sur un transistor MOS, chaque MOS connecte une entrée (diagonale) avec un bus de sortie (horizontal) comme le montre la figure V.19.

ouerdani:Tue May 13 17:25:56 1986
figV19

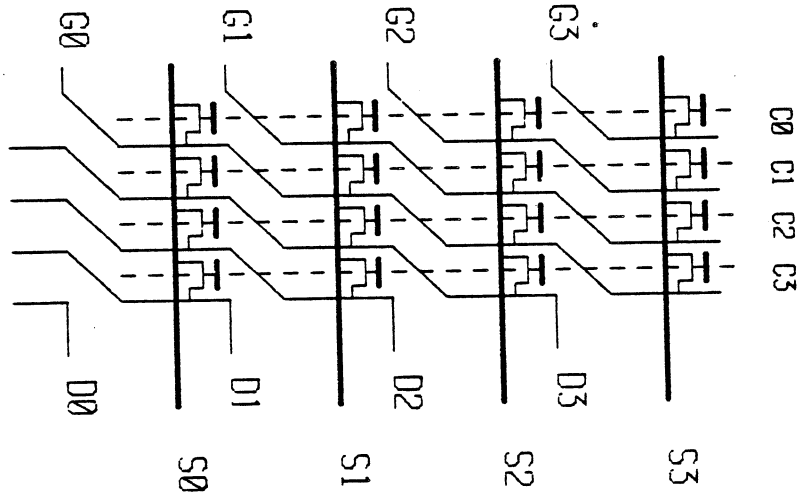


Figure V.19: décaleur matriciel 4x4.

Ce décaleur va donc se présenter topologiquiment de la manière suivante:

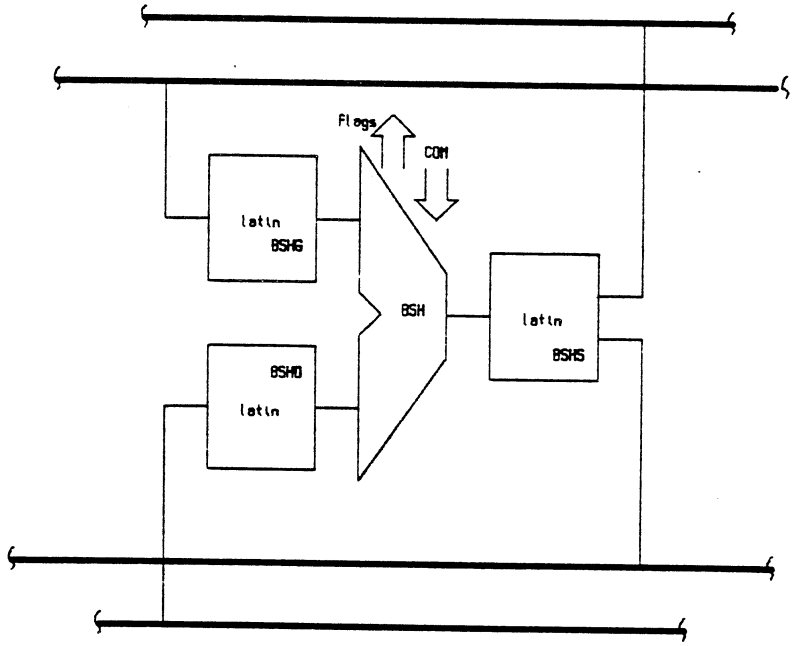


Figure V.20: représentation topologique du BSH

FONCTIONNALITE:

a) décalage à gauche de n positions (0<= n <=31):

Le résultat est obtenu en décalant le tampon d'entrée droit (IBSHD), vers la gauche (vers les poids Forts) de n positions; les n poids forts de IBSHD sont perdus; les n poids forts de IBSHG entrent en poids faibles du résultat.

a-1) Le décalage logique à gauche se fait en chargeant la valeur à décaler dans le tampon droit et des zéros dans le tampon gauche.

a-2) Le décalage arithmétique à gauche se fait aussi en chargeant la valeur à décaler dans le tampon droit et des zéros dans le tampon gauche. Toutefois le bit 0 du résultat (bit de signe) doit être le même que le bit 0 de la valeur en entrée (tampon droit).

b) Décalage à droite de n positions (0<= n <=31):

Le résultat est obtenu en décalant le tampon d'entrée gauche (IBSHG), vers la droite de n positions; les n poids faibles de IBSHG sont perdus; les n poids faibles de IBSHD entrent en poids forts du résultat.

b-1) Le décalage logique à droite se fait en chargeant la valeur à décaler dans le tampon gauche et des zéros dans le tampon droit.

b-2) Le décalage arithmétique à droite se fait en chargeant la valeur à décaler dans le tampon gauche et le bit de signe de la valeur à décaler dans tous les bits du tampon droit.

c) En plaçant la même valeur dans IBSHD et IBSHG, on effectue une rotation sur un mot de 32 bits.

Sur les 32 fils de commande de décalage, un seul d'entre eux doit être à l'état haut pendant la période dans laquelle le décalage a lieu. Si les sorties du décaleur sont préchargées de la même manière

que l'étaient les entrées (diagonales), les transistors interrupteurs constituant la matrice de décalage doivent forcer à l'état bas les sorties du décaleur seulement quqnd l'entrée correspondante est à l'état bas. Pour ce noyau, on minimise le retard du réseau de décalage et on utilise efficacement les possibilités technologiques.

DESCRIPTION DES COMMANDES BSHCi (i variant de 0 à 31):

Si la commande BSHCi est active alors le bit j du résultat Rj vaut:

- pour $0 \leq j \leq i-1$, Rj = IBSHG(j + 32 -i)
- pour $i \leq j \leq 31$, Rj = IBSHD(j-i)

Le tableau suivant montre pour chaque commande le nombre de positions de décalage effectués à droite et à gauche

commande Ci	nbre de décalages a droite	nbre de décalages a gauche
0	0	32
1	1	31
2	2	30
i	i	32 - i
30	30	2
31	31	1

TIMING:

Le chargement des tampons se fait en H3, la precharge du réseau de décalage en H0, décalage en PHI0 et H1, chargement du tampon de sortie en PHI1, sortie du résultat en PHI2.

CHAPITRE VI

ETUDE ET VALIDATION DES COMPOSANTS

VI - ETUDE ET VALIDATION DES COMPOSANTS

VI.1. REPRESENTATION ELECTRIQUE DES CIRCUITS

Pour simuler le comportement électrique dynamique d'un circuit, on le représente par son schéma électrique: c'est un réseau d'interconnections entre éléments primitifs (transistors, capacités, résistances, sources de tension, etc ...). Un simulateur électrique est une sorte d' "oscilloscope informatique", permettant de visualiser l'évolution dans le temps de divers paramètres (courant, tension,...) associés à des points du réseau.

Un tel simulateur électrique est précieux par la précision de la modélisation qu'il effectue (en supposant que ses paramètres propres ont été bien "calibrés" sur la technologie de fabrication). Ses inconvénients principaux sont le temps et la mémoire nécessaire à de telles simulations: quelques minutes de calcul pour une dizaine de transistors, et quelques heures dès la centaine sur un mini calculateur. Grâce à des percées récentes dans le domaine informatique, la simulation électrique de quelques milliers de transistors devient possible sur les gros ordinateurs. Simuler à ce niveau de détail 100.000 éléments actifs semble aujourd'hui hors d'atteinte.

Le simulateur électrique est pourtant le seul outil utilisable dans la mise au point des circuits intégrés. Comme une simulation exhaustive est impossible, la mise au point de circuit VLSI entier se révèle de plus en plus difficile. Les méthodes de CAO classiques ont beaucoup de mal à suivre les progrès technologiques.

La vérification de cohérence entre les représentations géométriques et électriques du circuit peut être automatisée: un extracteur de circuit VLSI construit le schéma électrique à partir du plan des masques (layout) et des règles technologiques, un comparateur est un outil de vérification d'homomorphisme entre les deux graphes et qui est

parfois utile dans la mesure ou l'un des deux shémas constitue une référence.

VI.2. OUTILS EMPLOYES

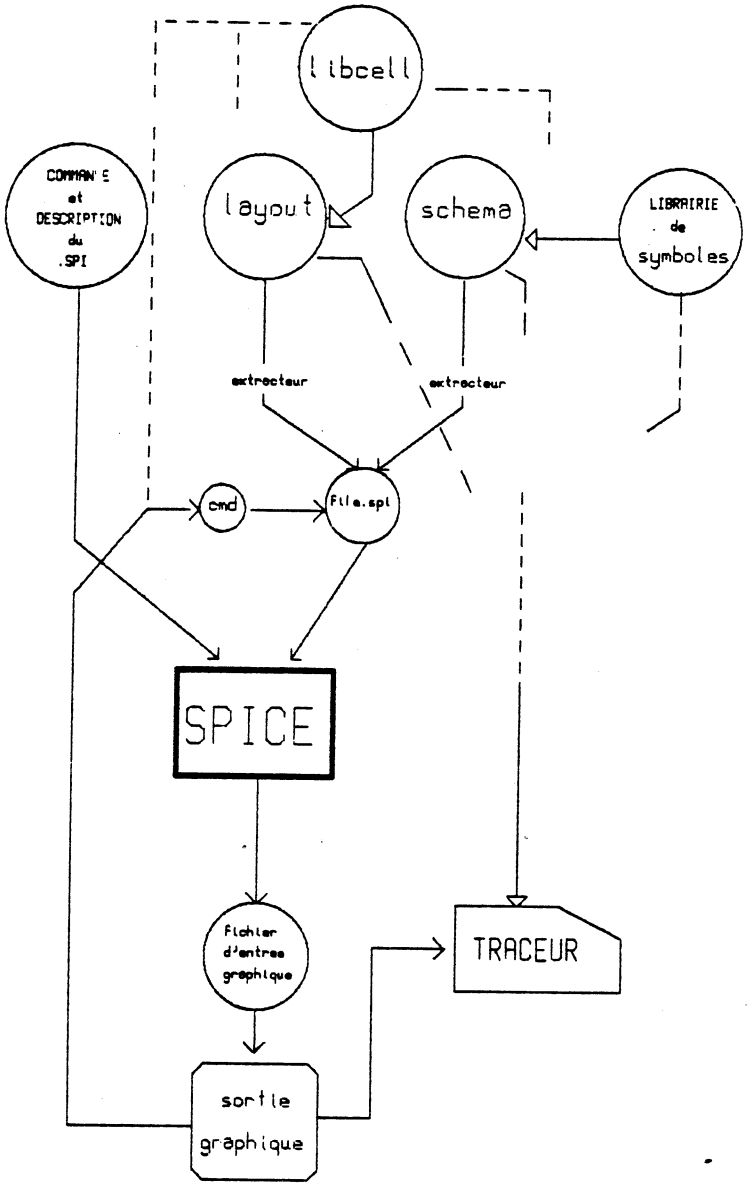
Les programmes de vérification des règles de dessin (ou DRC: Design Rule Checking) permettent au concepteur de s'assurer automatiquement de la conformité du dessin de masques avec les règles technologiques. Une telle vérification est de nature statique : elle s'assure que le circuit pourra bien être réalisé conformément aux spécifications de ses masques, en ne donnant aucune indication sur son comportement dynamique . On peut dire que cette vérification joue, pour les descriptions de circuits, le rôle de l'analyse lexicale dans les compilateurs de langages.

Pour mettre en valeur le comportement dynamique d'un circuit, on est amené à utiliser des simulateurs électriques et logico-temporels. Nous allons en présenter quelques-uns que l'on a utilisé.

VI.2.1. LE SIMULATEUR ELECTRIQUE SPICE

Pour simuler électriquement les différents composants décrits dans le chapitre V, nous utilisons le simulateur électrique Spice.

Ce simulateur peut être utilisé de différentes manières comme le montre le diagramme suivant:



ouerdan1:Tue May 13 17:31:24 1986
figVI1

Figure VI.1: diagramme d'utilisation de Spice

Ce diagramme montre que suivant le cas, le concepteur pourra lancer sa simulation à partir de fichiers différents en utilisant l'une ou l'ensemble des méthodes suivantes:

- soit qu'on parte de la méthode classique qui consiste à décrire le

circuit à simuler sur un fichier (file.spi) grace à un éditeur de texte et lancer par la suite Spice avec les options adéquates. Après des résultats satisfaisants de simulation, le concepteur pourra implanter son circuit à l'aide d'un éditeur graphique;

- soit partir d'un plan de masque et extraire par programme le fichier spice auquel il faut, sous l'éditeur de texte, rajouter les commandes pour le simulateur;

- soit partir d'un schéma électrique édité sous un éditeur de schéma qui utilise des symboles dessinés sous un éditeur de symbole, pour extraire un fichier spice auquel il faut adjoindre, sous l'éditeur de texte, les commandes désirées.

Les résultats de la simulation Spice, indépendamment de la méthode utilisée, sont stockés dans un fichier qui est ensuite pris en entrée d'un utilitaire de sortie graphique qui permettra la visualisation de l'évolution d'un certain nombres de noeuds du réseau simulé dans le temps. Cet utilitaire permet également des visualisations partielles de l'ensemble des noeuds en rendant visible seulement les noeuds désirés. Il permet aussi les "zooms" ou définition d'une fenêtre pour lire, d'une façon précise, les coordonnées d'un point. Il permet également de faire des sorties graphiques sur imprimantes électrostatiques.

VI.2.2. SIMULATEUR SWITCH

Une approche que nous avons utilisée aussi est la simulation logico-temporelle grace au simulateur "switch". Ce simulateur modélise les chaînes de transistors en chaînes RC et permet ainsi de donner le temps de propagation de l'information entre l'entrée et la sortie d'un réseau de "switchs"; ceci permet de déterminer les chaînes critiques qu'il faut regarder de plus près avec Spice. Ce simulateur utilise soit des schémas édités sous l'éditeur de layout et dont on extrait les caractéristiques électriques par l'extracteur, soit des schémas créés sous l'éditeur de schéma. Les résultats sont fournis soit

sous forme de tableaux de vecteurs dont il faut interpréter la signification, soit sous forme de chronogrammes de temps. Il est à noter que ce simulateur est utilisé avant tout pour des sous ensembles plus importants (quelques milliers de transistors).

VI.3. ETUDE DE MECANISMES DE PRECHARGE DE BUS

VI.3.1. EVALUATION DE LA CAPACITE DE BUS

Les éléments qui rentrent en compte dans le calcul de la capacité d'un bus en métal2 sont:

- la capacité du rail en métal2 (ou M2) composée par la somme de sa capacité de fond avec sa capacité de bord et ceci en fonction de la longueur du rail;

- les capacités de croisement;

- les capacités de drains des transistors de sélection des points mémoire et des amplificateurs de bus connectés sur celui-ci.

VI.3.2. ETUDE DE LA PRECHARGE

Il s'agit de dimensionner les transistors, du système de précharge, qui vont faire monter à 4.2 volts le bus qui est descendu à 0 volt lors du cycle précédent. Les phases de précharge sont H0 et H1 du cycle machine. Les premiers schémas de principe qui vont servir à précharger les bus vont être donnés par les figures suivantes.

PREMIER MECANISME

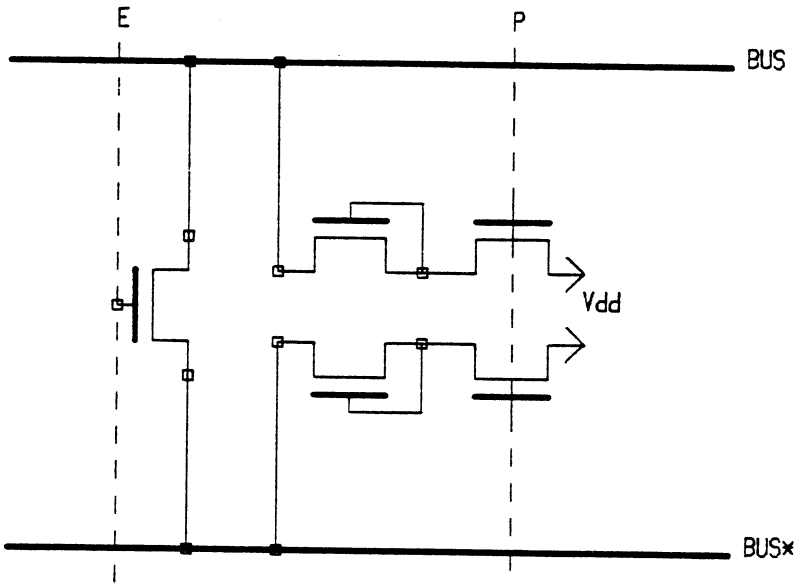


Figure VI.2: système de précharge de bus à transistors N

On distingue les deux transistors type N de précharge à commande P qui amènent du courant vers les bus (BUS et BUS*), les deux transistors "abaisseurs de tensions" avec leur grille/drain court-circuités; ces deux transistors vont ramener la tension de bus au alentour de 4.2 volts (Vs <=Vg-Vt); enfin, un transistor commandé par un signal E et qui court-circuite le bus et le bus complémenté. Ce signal va réaliser l'égalisation de la tension des deux bus.

Les deux commandes P (Précharge) et E (Egalisation) vont être réparties de la manière suivante:

- durant la phase H0, c'est la commande P qui est au niveau haut et E est au niveau bas;

- durant la phase H1, E est activé et P est à l'état bas.

Le modèle qui va être utilisé par la suite pour simuler cette précharge est représenté par la figure VI.3:

ouerdani:Tue May 13 17:32:37 1986
figVI3

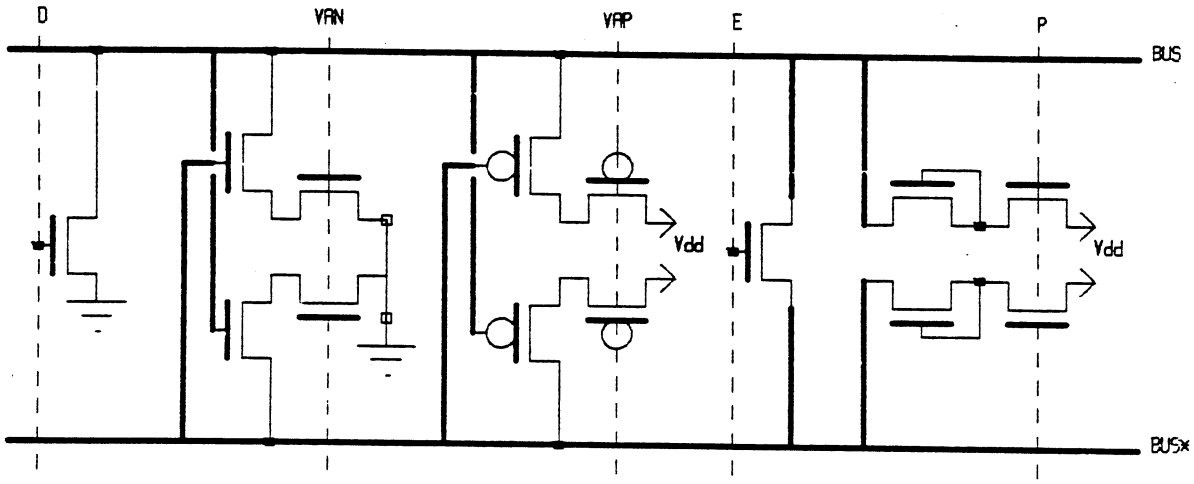


Figure VI.3: Modèle de la simulation de bus

Le transistor commandé par le signal D (comme Décharge) va servir de modèle de point mémoire pour la lecture d'un "0" par le bus et les deux amplificateurs différentiels vont intervenir au moment de cette lecture pour ramener le bus complémenté à "1".

Vu la capacité importante du bus à précharger, nous dimensionnons les transistors de précharge , le transistor d'égalisation et les "résistances" de manière à ce qu'on arrive à le précharger dans les deux phases.

Le résultat de la simulation Spice est donné par la figure VI.4. Nous remarquons qu'au niveau du bus (noeud 1), la précharge se fait relativement lentement pour se terminer à la fin de la phase H0 à une tension voisine de 2.5 volts et, durant la phase H1 d'égalisation, le bus atteint à peine 3.7 volts et non les 4.2 volts espérés; en plus, les deux bus ne s'égalisent pas réellement ce qui nécessite un surdimensionnement du transistor de court-circuit, mais nous pensons qu'il ne faut pas aller plus loin puisque les transistors sont assez gros et vont occuper une grande place lors de leur implantation, ce qui nous conduit à chercher d'autres solutions.

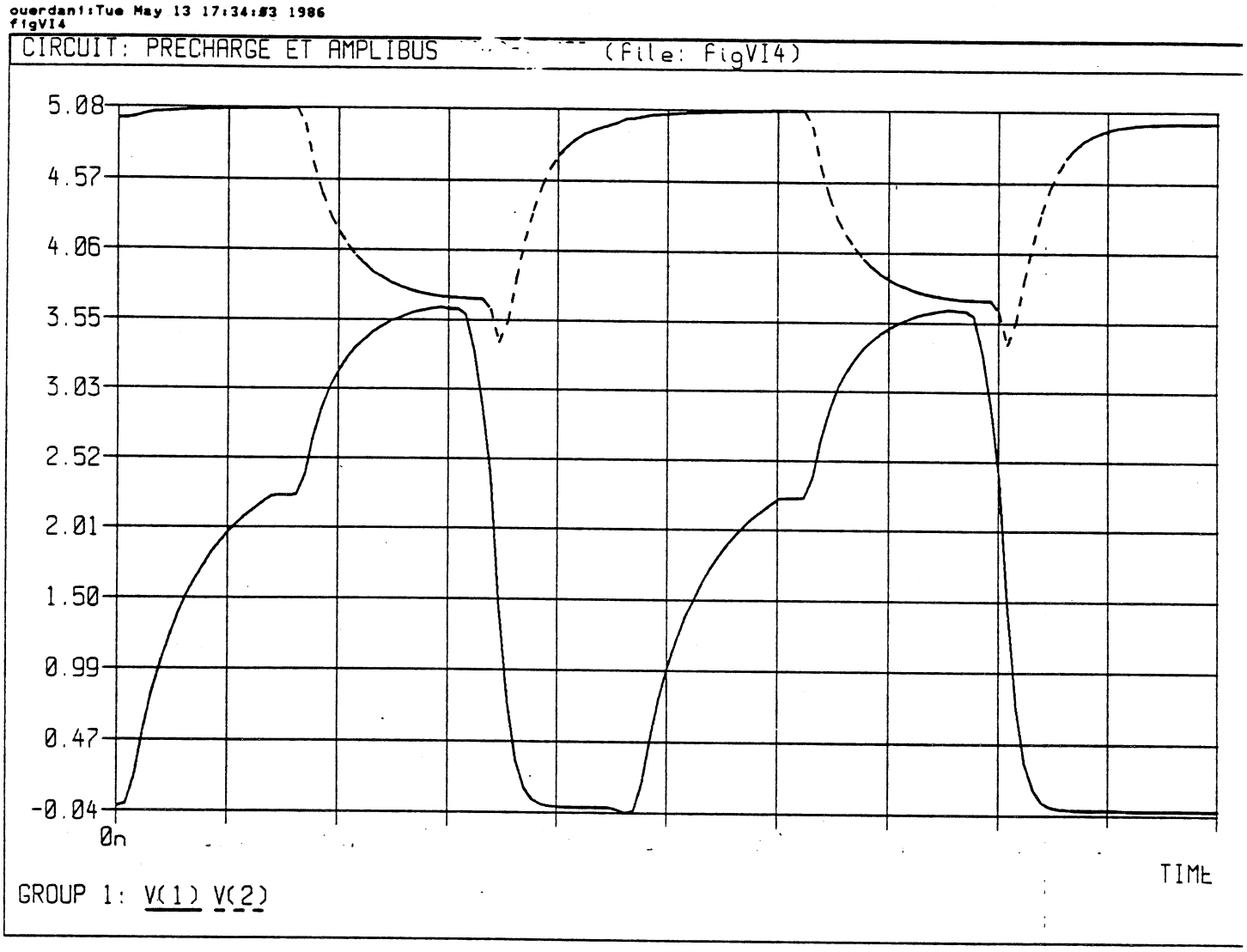


Figure VI.4: courbes de la simulation Spice du mécanisme de précharge à transistors N

DEUXIEME MECANISME

Regardons maintenant ce qui se passe avec le montage suivant:

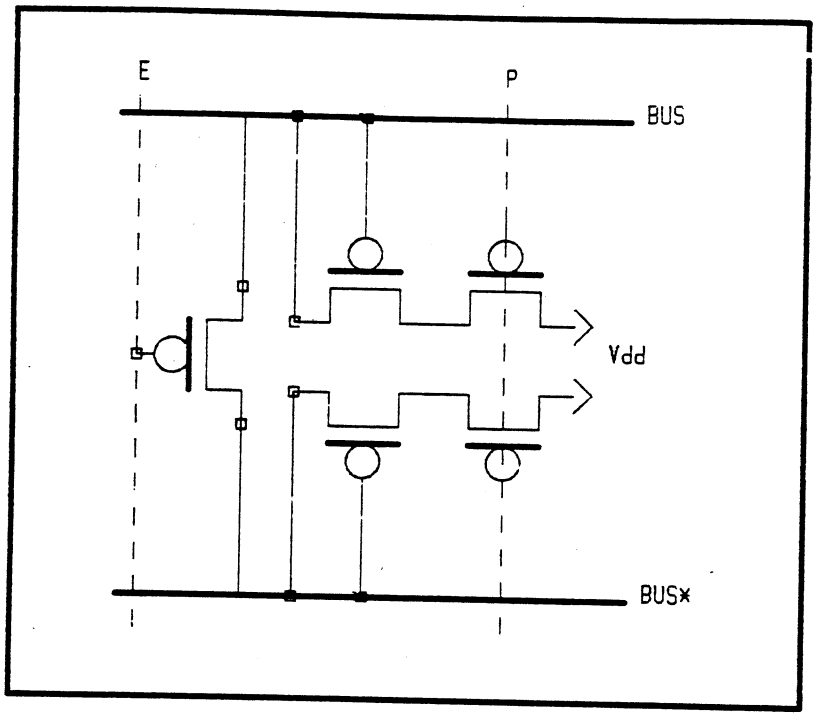


Figure VI.5: système de précharge de bus à transistors P

On voit l'intérêt de ce mécanisme par rapport au premier par le fait que seul le transistor dont le drain est relié au bus de niveau bas est conducteur, l'autre étant bloqué puisque sa grille se trouve au niveau haut (bus complémenté). On refait la même simulation résultant du schéma de la figure VI.6 en adoptant les mêmes dimensions que précédemment.

ouerdani:Tue May 13 17:35:10 1986
figVI6

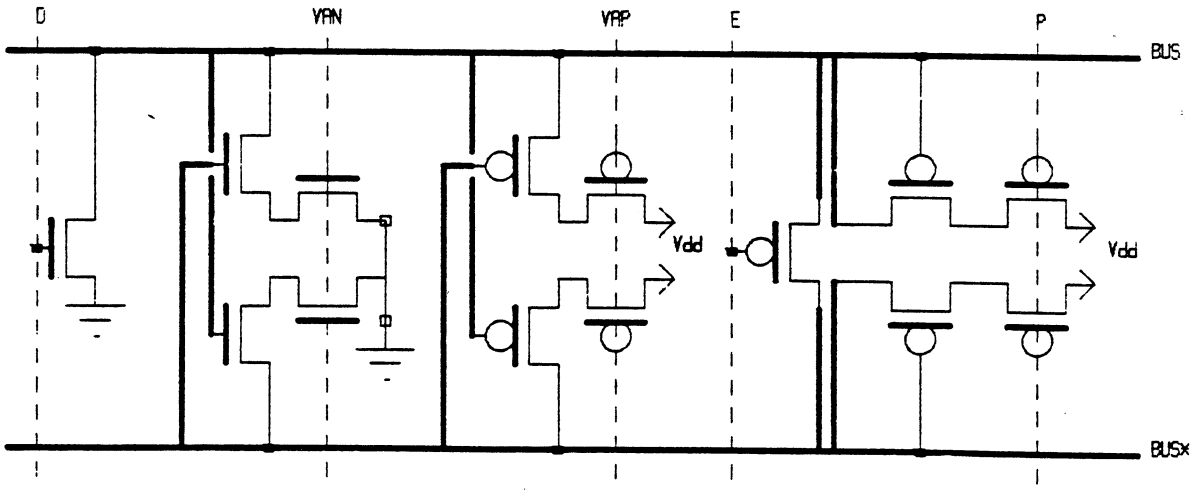


Figure VI.6: Modèle général de la simulation de bus

Après étude du résultat (figure VI.7) de la simulation Spice, nous constatons que le bus (noeud 1) monte plus que précédemment pour atteindre 3.4 volts avant la fin de la première phase (au lieu de 2.5 volts pour l'autre montage!) et que les deux bus s'égalisent parfaitement à 4.2 volts avant la fin de la deuxième phase.

On ramène les dimensions du transistor d'égalisation à une valeur inférieure et on constate que le résultat reste toujours bon: l'égalisation à 4.2 volts se fait très rapidement(fig. VI.8). Nous adoptons ces géométries et la figure VI.9 montre l'implantation de ce système de précharge.

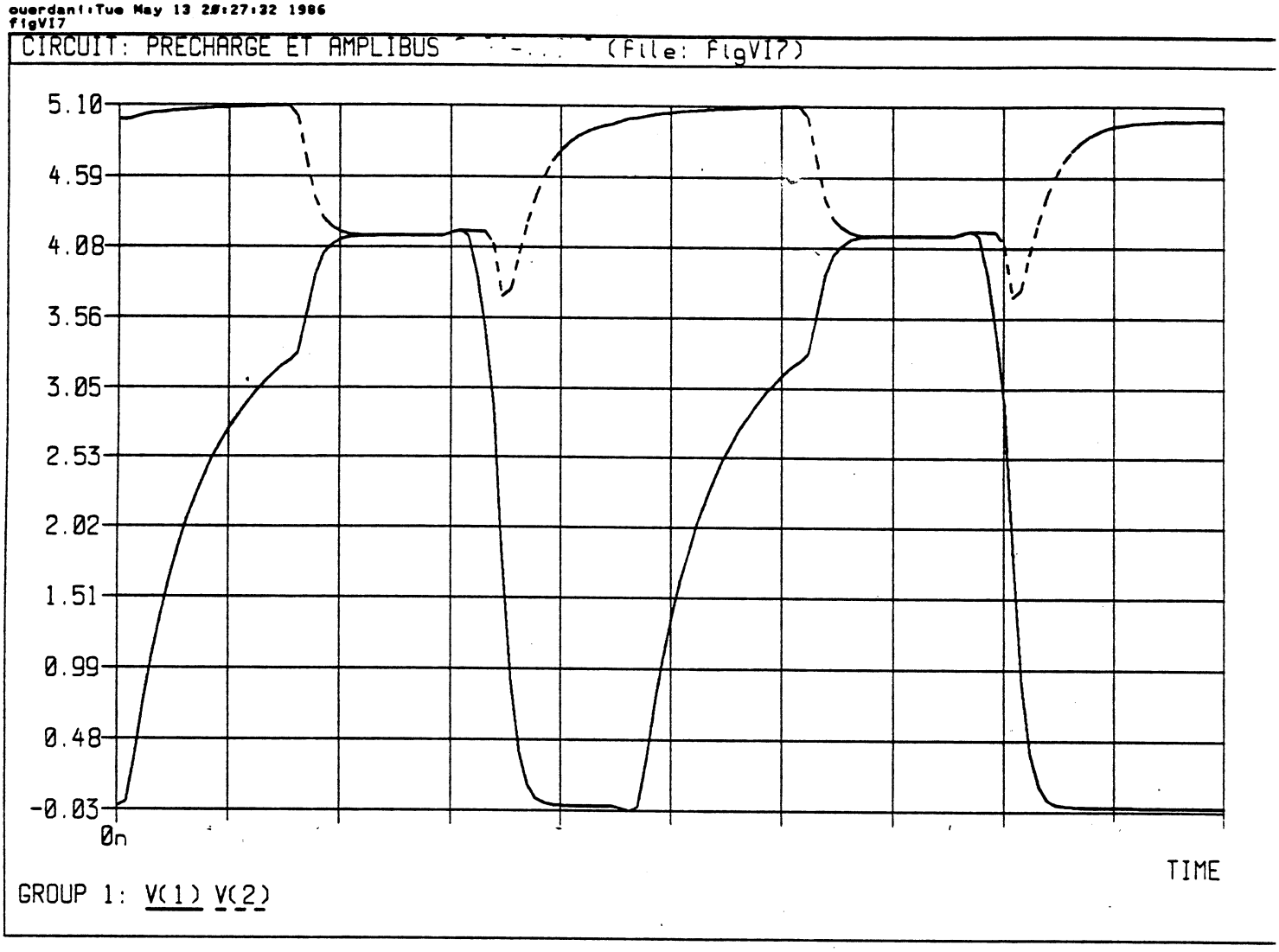


Figure VI.7: simulation du mécanisme de précharge

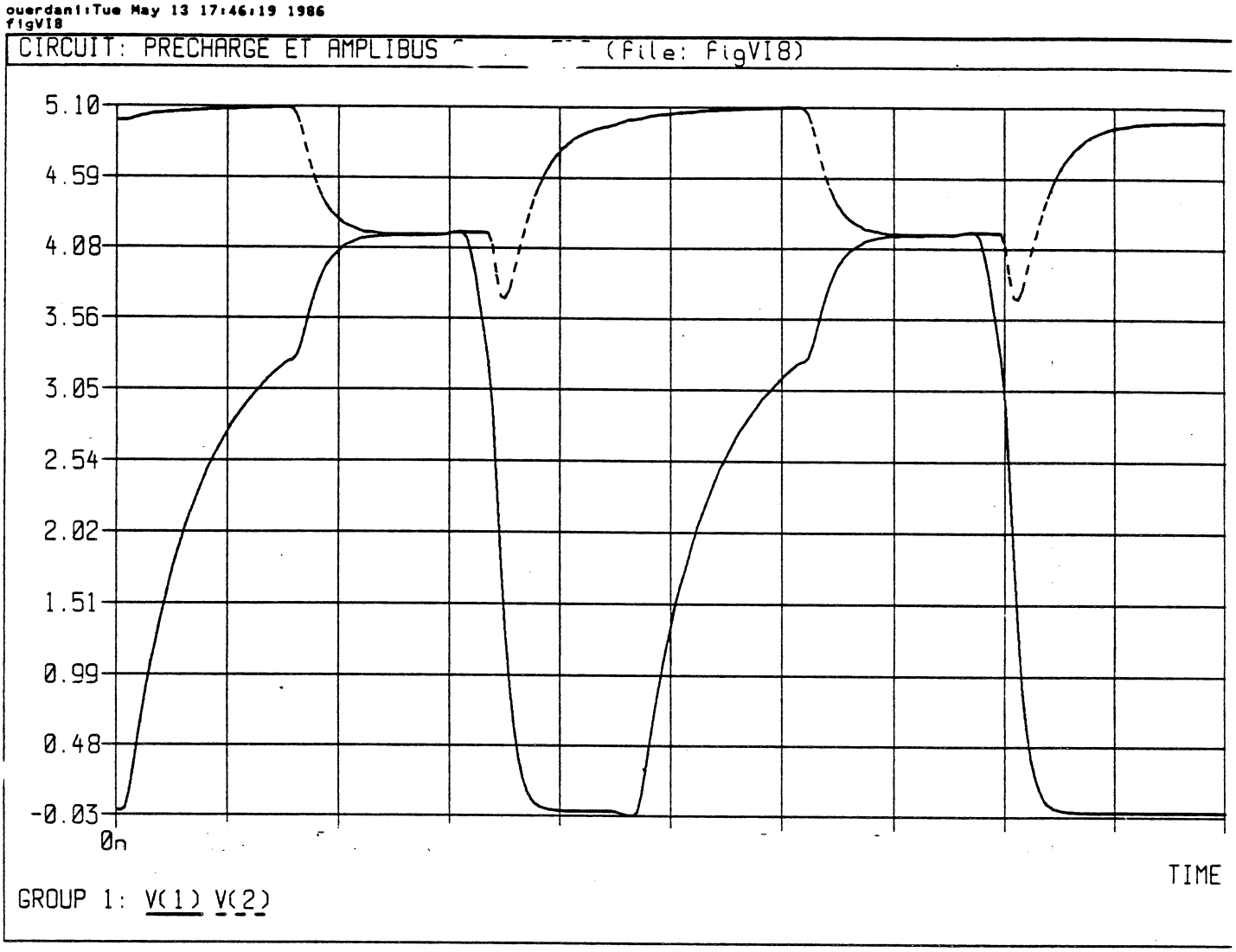


Figure VI.8: Amélioration du mécanisme de précharge

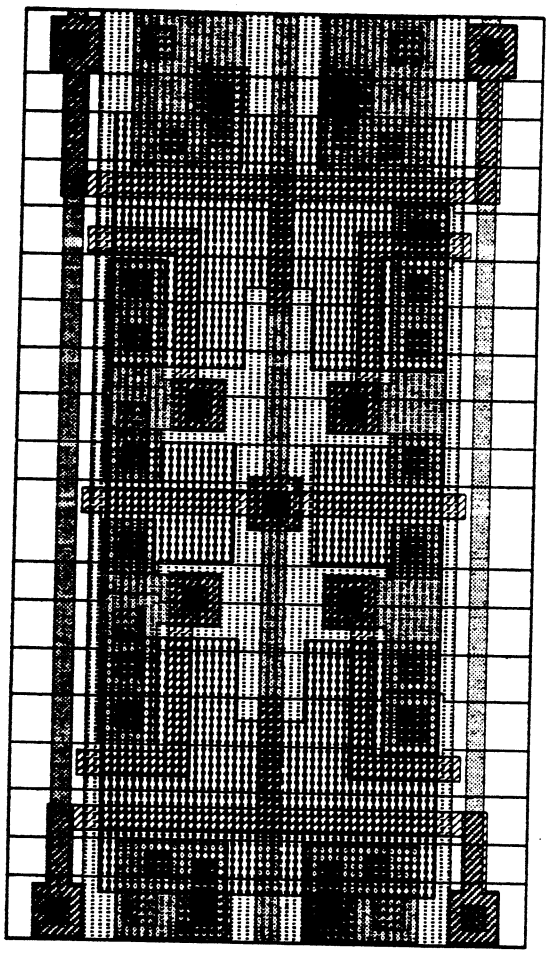


Figure VI.9: implantation du mécanisme de précharge

VI.4. ETUDE DU POINT MEMOIRE

Nous allons étudier le comportement d'un point mémoire double accès sur deux bus notés BUSA et BUSB et dont la représentation électrique est donnée par la figure VI.10:

ouerdani:Tue May 13 17:47:06 1986
figVI10

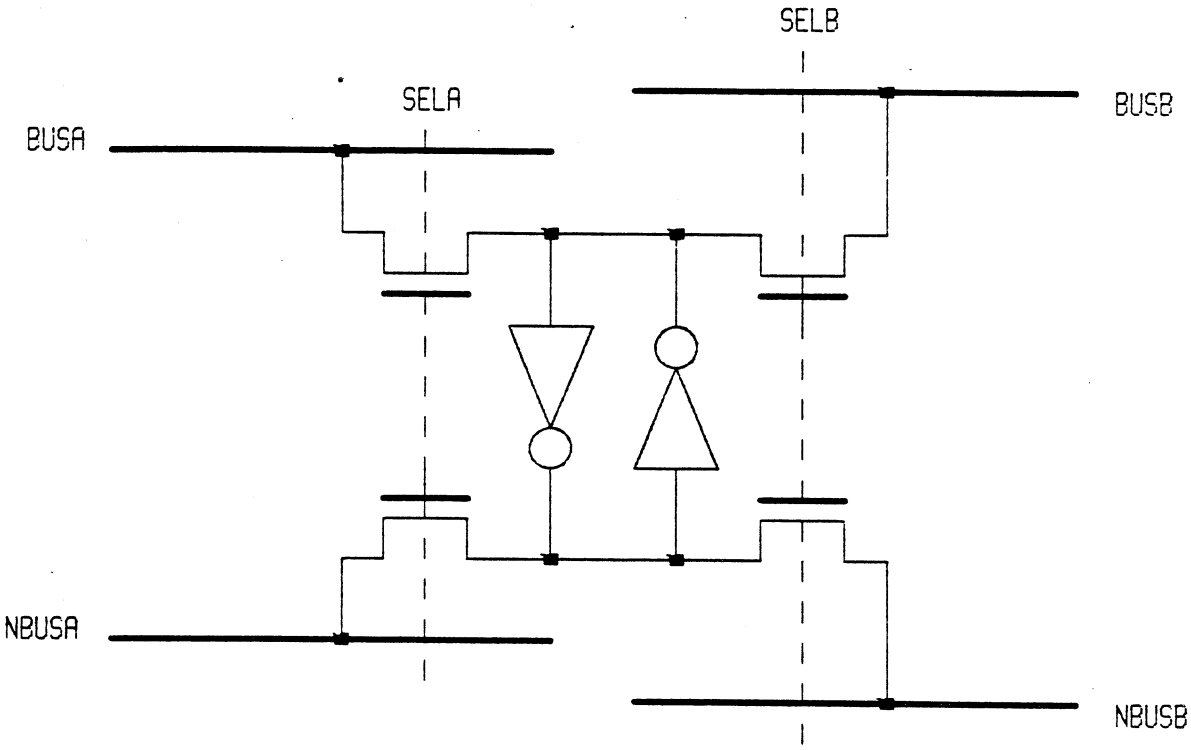


Figure VI.10: Schéma éléctrique d'un point mémoire double accès

VI.4.1. ETUDE DE LA LECTURE

Il s'agit de dimensionner les transistors du point mémoire pour faire basculer le bus précédement chargé à 4.2 volts vers 0 volt. Comme la capacité des bus est importante par rapport aux capacités d'entrée (dans un rapport 100) des transistors d'inverseurs constituant le point mémoire, nous donnerons des valeurs importantes aux transistors des inverseurs rebouclés pour aider encore plus cette décharge (lecture).

Les dimensions des transistors du point mémoire sont choisies aussi bien pour le type N que pour le type P. Pour les transistors de passage on prend des valeurs relativement faibles. A ce sujet, il est conseillé que les paramètres géométriques soient tels que toutes les portes inverseuses du point mémoire soient équilibrées. Le schéma général servant à notre simulation est donné à la figure VI.11:

ouerdani:Tue May 13 17:47:42 1986
figVI11

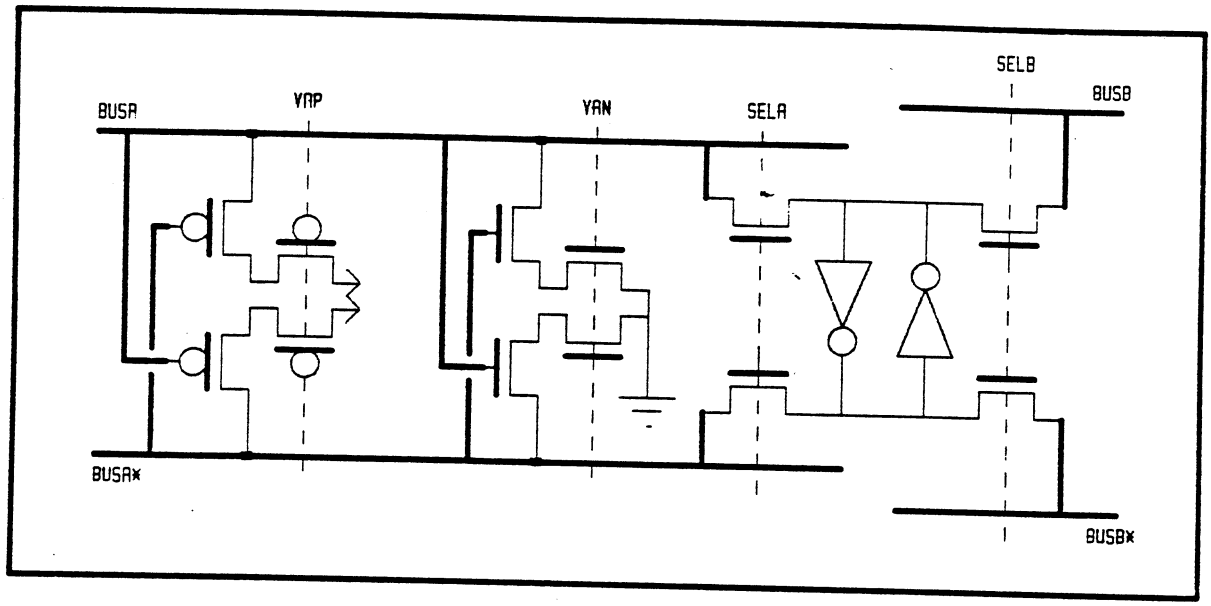


Figure VI.11: Schéma de simulation de L/E du point mémoire

Les signaux VAN et VAP sont les activations (ou Validation) des amplificateurs N et P respectivement.

A l'initialisation, BUSA et BUSA* sont au niveau haut, BUSB est au niveau haut et BUSB* au niveau bas, de telle manière que les valeurs des BUSB et BUSB* soient à l'opposé de celles du point mémoire en phase d'écriture.

Le résultat de cette simulation est donné à la figure VI.12 où on peut remarquer que la lecture se passe très bien puisque le bus BUSA bascule rapidement (noeud 8) tandisque le bus complémenté se decharge d'environ 1 volts avant de se charger et reprendre la valeur de 5 volts. Ce phénomène résulte de l'adjonction des amplificateurs différentiels N et P. En effet, après la validation de l'amplificateur différentiel N (VAN ="1"), celui-ci va tirer à la masse le bus à décharger (BUSA), à travers deux de ses transistors, donc il va avoir un rôle d'accélérateur de cette chute de tension, mais en contrepartie, le bus BUSA* restant à "1" va suivre momentanément cette décharge à travers l'autre couple de transistors de l'amplificateur en question, jusqu'au moment où le bus BUSA va bloquer l'un de ces deux

transistors pour qu'il s'arrête de se décharger. Sa valeur aura donc tendance à rester en dessous des 4.2 volts et pour remédier à cet inconvénient, on a pris soin d'activer l'amplificateur différentiel P, immédiatement après l'activation de l'ampli N, qui va tirer ce bus à "1" (V9).

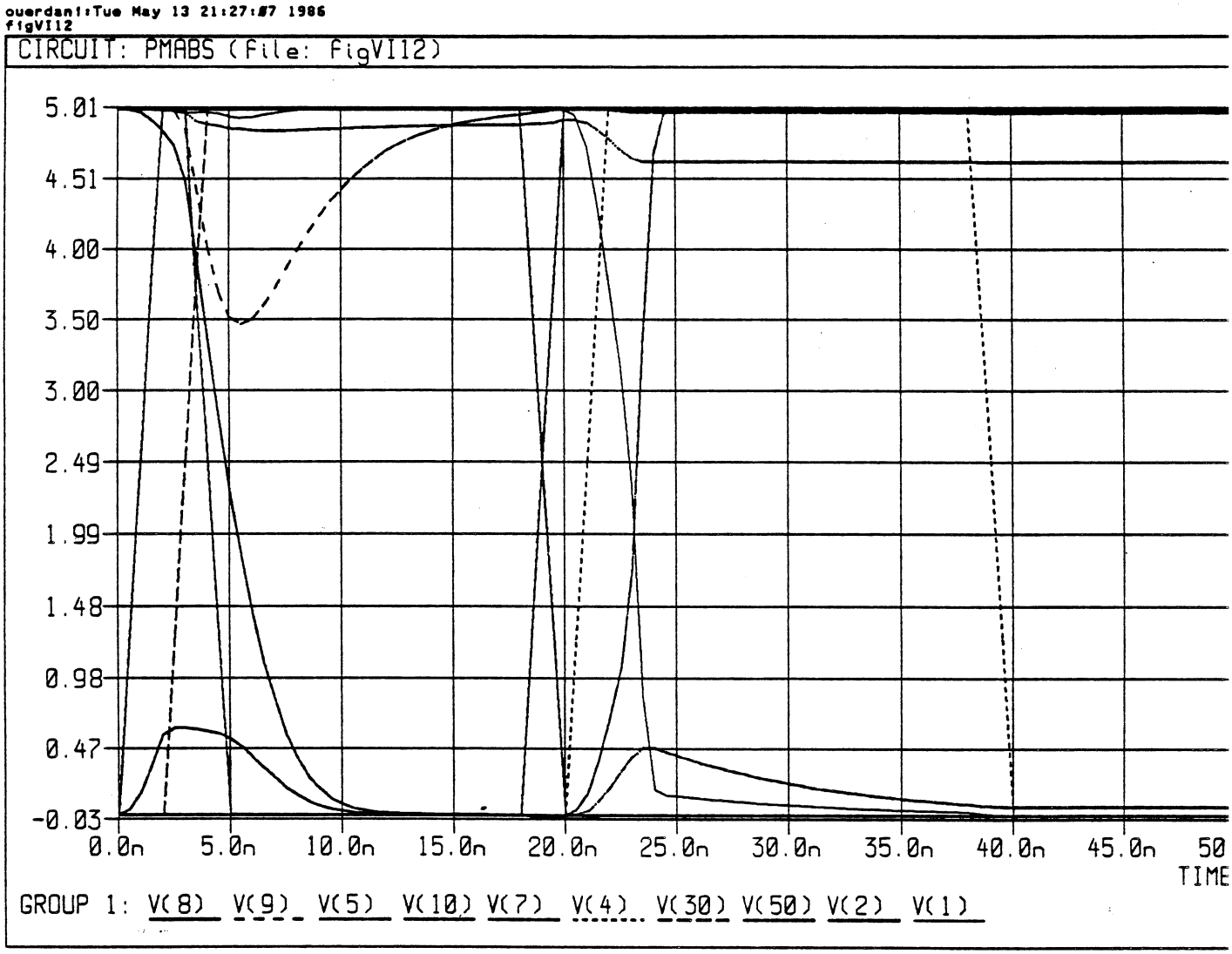


Figure VI.12: Simulation Spice du point mémoire

VI.4.2. ETUDE DE L'ECRITURE

Le cas de l'écriture de la valeur d'un bus dans le point mémoire n'est pas tellement primordial du fait que les bus ont une capacité

importante vis-à-vis de celle du point mémoire, par conséquent, et le résultat de la simulation le montre, le bus ne trouve aucune difficulté à "imposer" sa valeur au point mémoire (noeud 2 et 1).

VI.4.3. IMPLANTATION

Après ces simulations, on a implanté les points mémoires suivant une stratégie globale en tenant compte de l'environnement dans lequel ils vont être intégrés (surtout l'alignement et le partage de la frontière avec les cellules avoisinantes). Nous présentons en figure VI.13 l'implantation d'un point mémoire double accès.

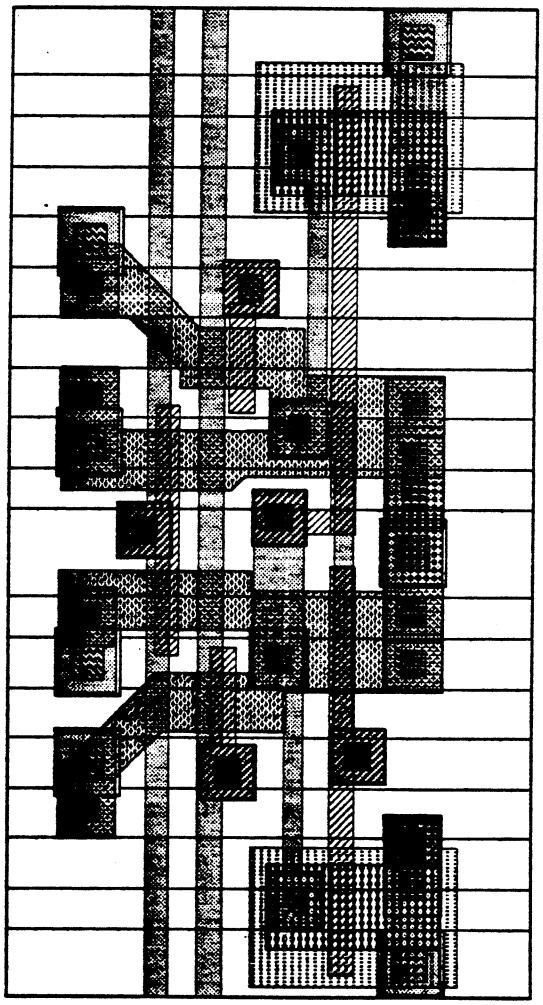


Figure VI.13: point mémoire double accès

VI.5. COMPORTEMENT DES LATCHS

Après l'étude des points mémoires, nous allons nous pencher sur l'étude de quelques points de mémorisation particuliers: les "latchs".

Nous allons voir la réaction de ces latchs en lecture et en écriture d'un "0" ou d'un "1".

VI.5.1. LATCH1

Nous commençons par le premier type donné par la figure suivante:

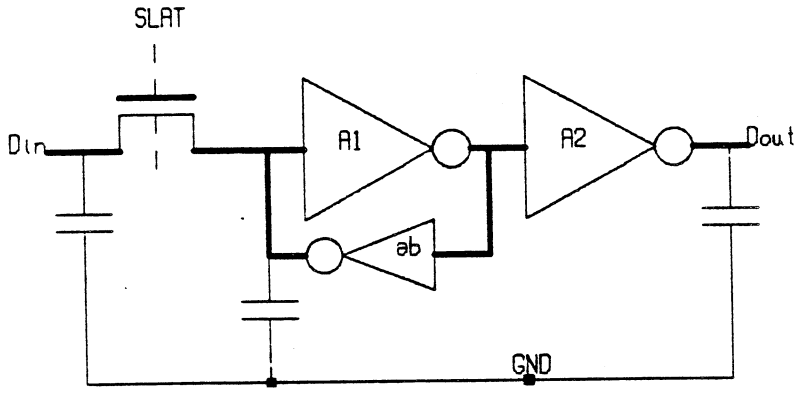


Figure VI.14: latch à double inverseurs rebouclés

On met une capacité à sa sortie qui va représenter une entrée d'un opérateur et une capacité à son entrée qui va modéliser la capacité d'un bus de donnée.

Le schéma de simulation que nous allons adopter est le suivant:

ouerdani:Tue May 13 18:00:10 1986
figVI15

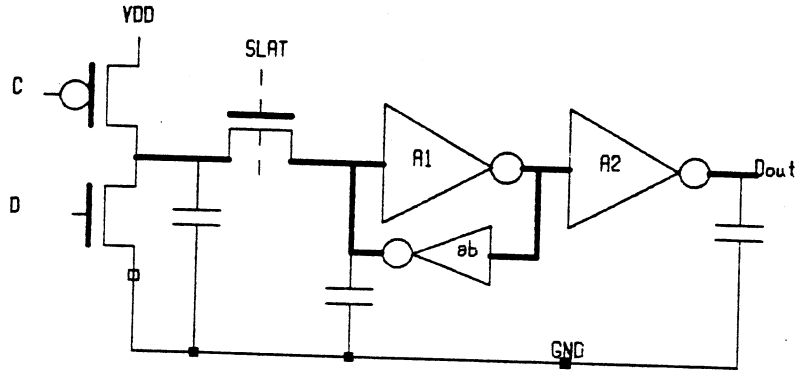


Figure VI.15: schéma de simulation d'un latch

SLAT est la commande de sélection du latch en écriture;
C est la commande de chargement d'un "1" dans le latch;
D est la commande de chargement d'un "0" dans le latch.

Le résultat de la simulation Spice montre que la transmission d'un "1" (i-e passage de 5 volts entre le moment de la sélection du latch et la sortie après amplification) et la transmission d'un "0" se font rapidement. Les mesures des temps de passage sont faits à 2.5 volts.

Le cas qu'on vient de voir est celui d'un latch rattaché à un bus; regardons maintenant ce qui se passe lorsque ces latchs ne sont pas rattachés à un bus, c'est le cas de registres à décalage par exemple (les latchs voient maintenant une faible capacité à leur entrée). Nous remarquons que la transmission d'un "1" se fait relativement bien, la transmission d'un "0" se fait rapidement. Si nous augmentons la capa d'entrée les temps de passage restent pratiquement inchangés. Diminuons cette capa, nous constatons que cette capa n'arrive pas à imposer sa valeur au latch pour qu'il bascule. De proche en proche on arrive à la limite de cette capa en dessous de laquelle on ne peut pas faire basculer le latch.

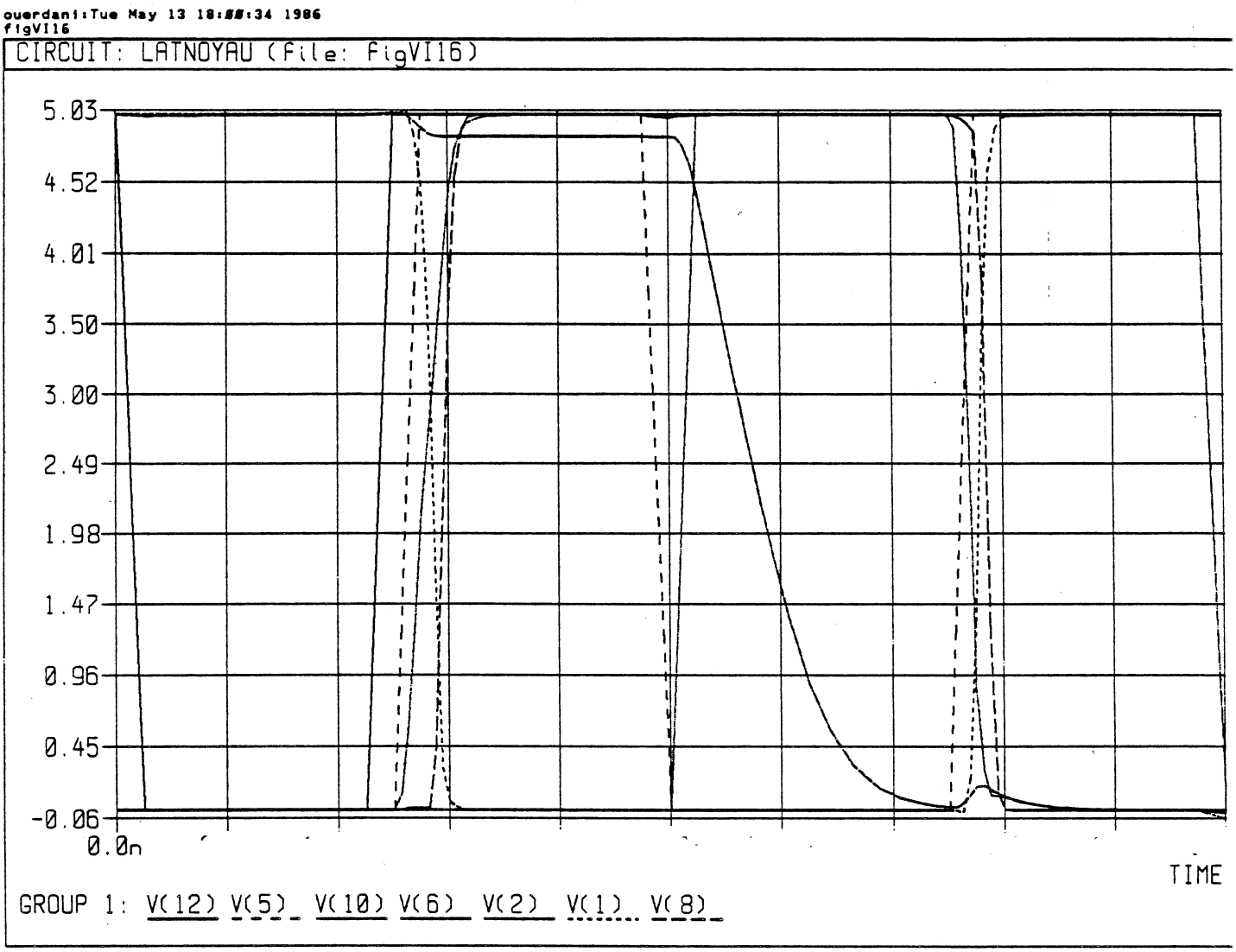
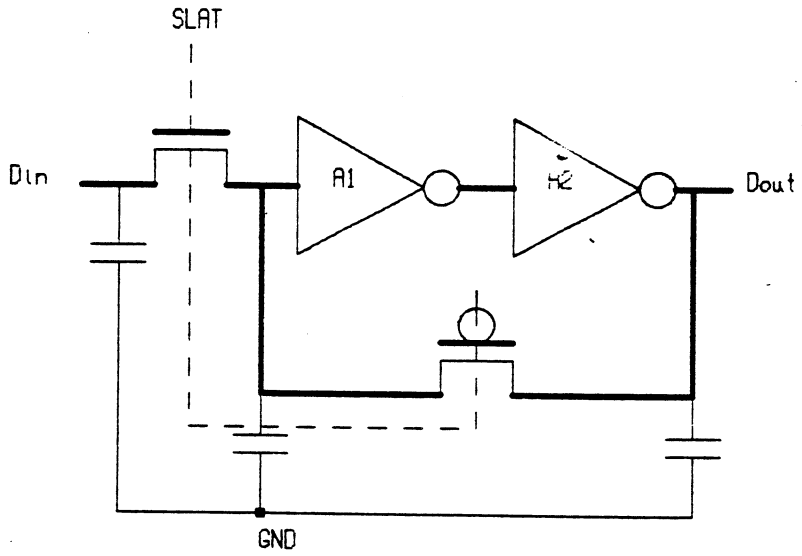


Figure VI.16: simulation de latch1

VI.5.2. LATCH2

Le deuxième latch simulé est donné par la figure V.17 suivante:

ouerdani:Tue May 13 18:00:21 1986
figVI17



Figre VI.17: latch à bouclage par un transistor P

L'étude du résultat de la simulation Spice nous montre que pour la transmission d'un "0" entre la sélection (V10) et la sortie (V8) à 2.5 et la transmission du "1" logique ont très peu changé par rapport au montage précédent.

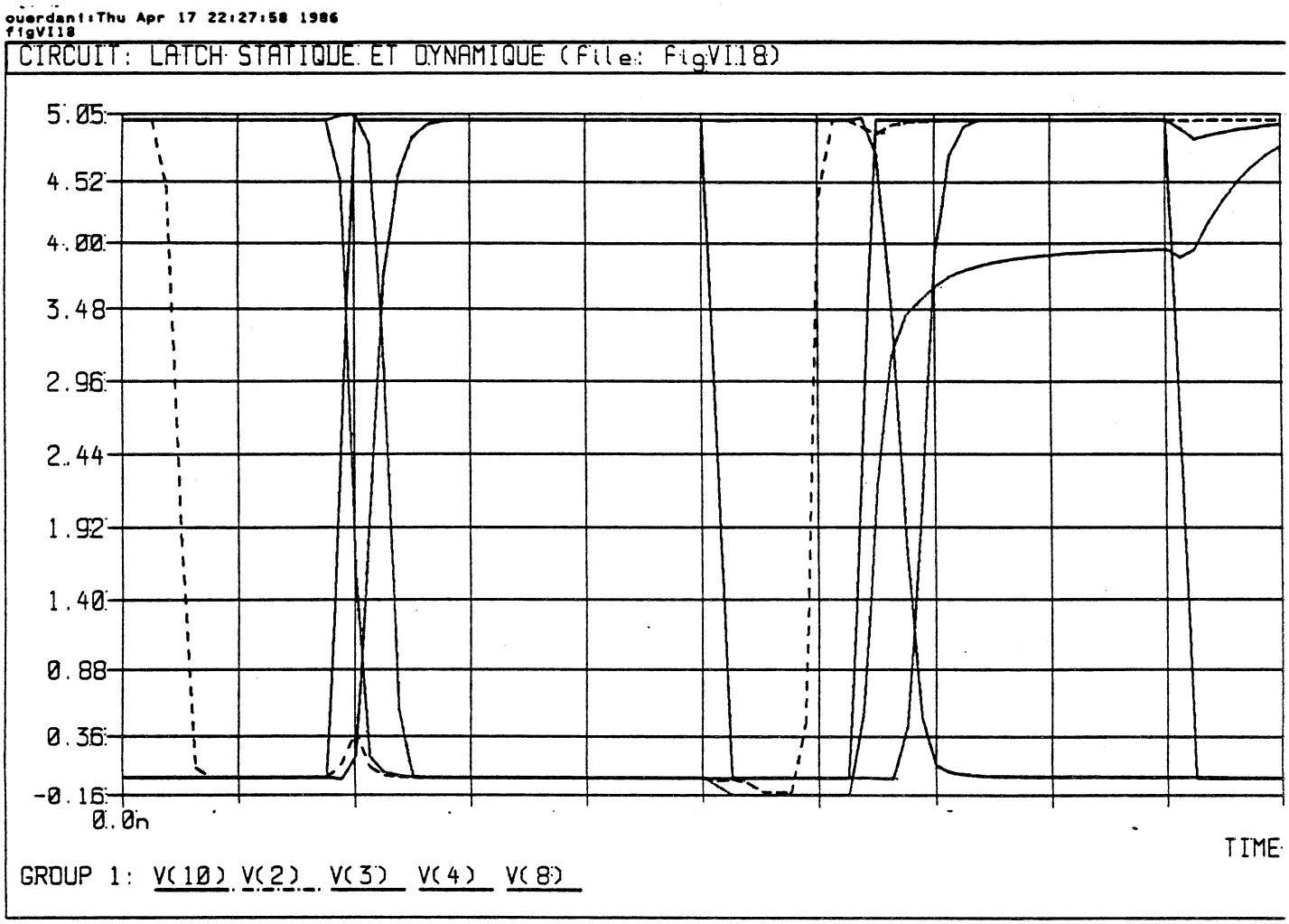


Figure VI.18: simulation de latch2

Rermarquons que cette simulation est faite dans les mêmes conditions que pour le latch1 (ceci pour faire des comparaisons entre les différents montages). Nous remarquons que l'entrée de ce latch est perturbée par un effet de diode dûe au bouclage par un transistor P. Un conflit surgit donc à ce niveau au passage de "0" à "1". Pour remédier à ce conflit, nous avons pensé rajouter un deuxième transistor de bouclage comme le montre la figure VI.19:

VI.5.3. LATCH3

ouerdani:Tue May 13 20:34:56 1986
figVI19

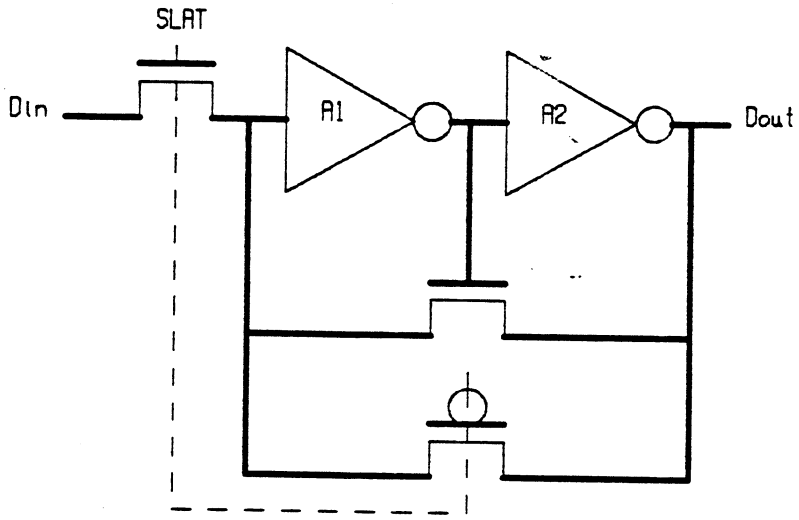


Figure VI.19: version "améliorée" du latch2

Là encore on voit apparaître ce problème de bouclage qu'on n'est pas arrivé à éliminer.

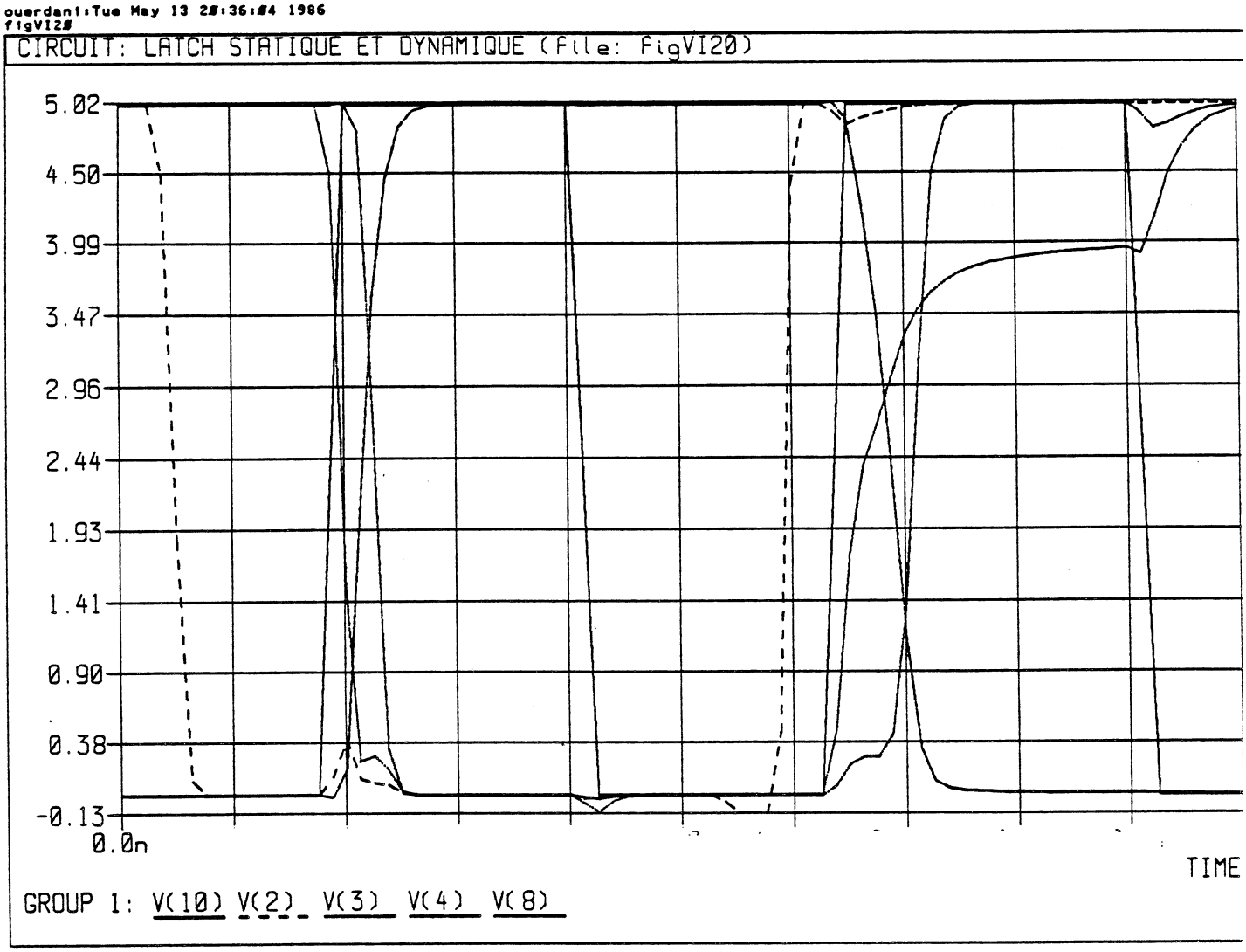


Figure VI.20: simulation de latch3

Conclusion: si on compare ces trois montages du point de vue performance et temps de réponse (à géométries égales), on est amené à adopter le latch1.

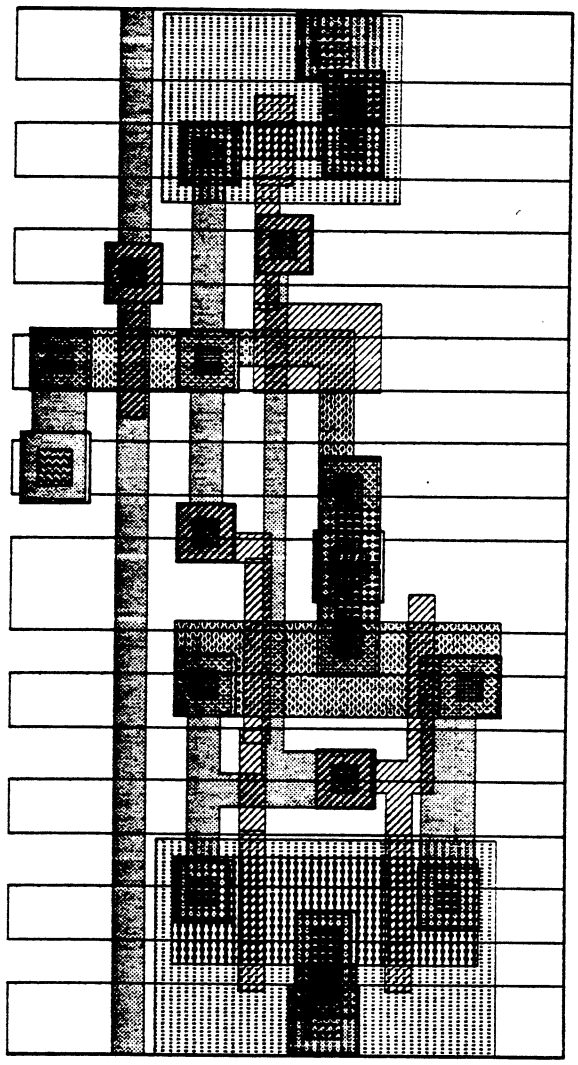


Figure VI.21: implantation de latch1

VI.5.4. LATCH4

Nous présentons une autre variante du latch précédent (latch4) qui est, cette fois, disymétrique vis-à-vis de la donnée. Son schéma est le suivant:

ouerdani:Tue May 13 20:37:12 1986
figVI22

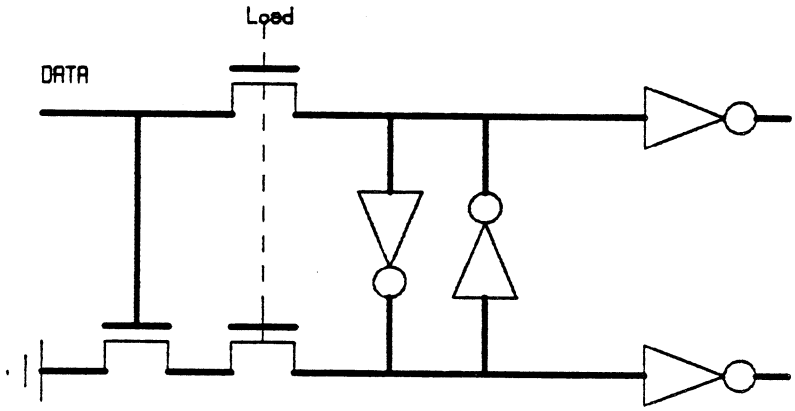


Figure VI.22: schéma électrique de latch5

Nous avons fait deux simulations dans les mêmes conditions que celles de latch4. On voit donc que le latch5 est très performant et conviendrait très bien pour le cas où on attaque deux bus à la fois.

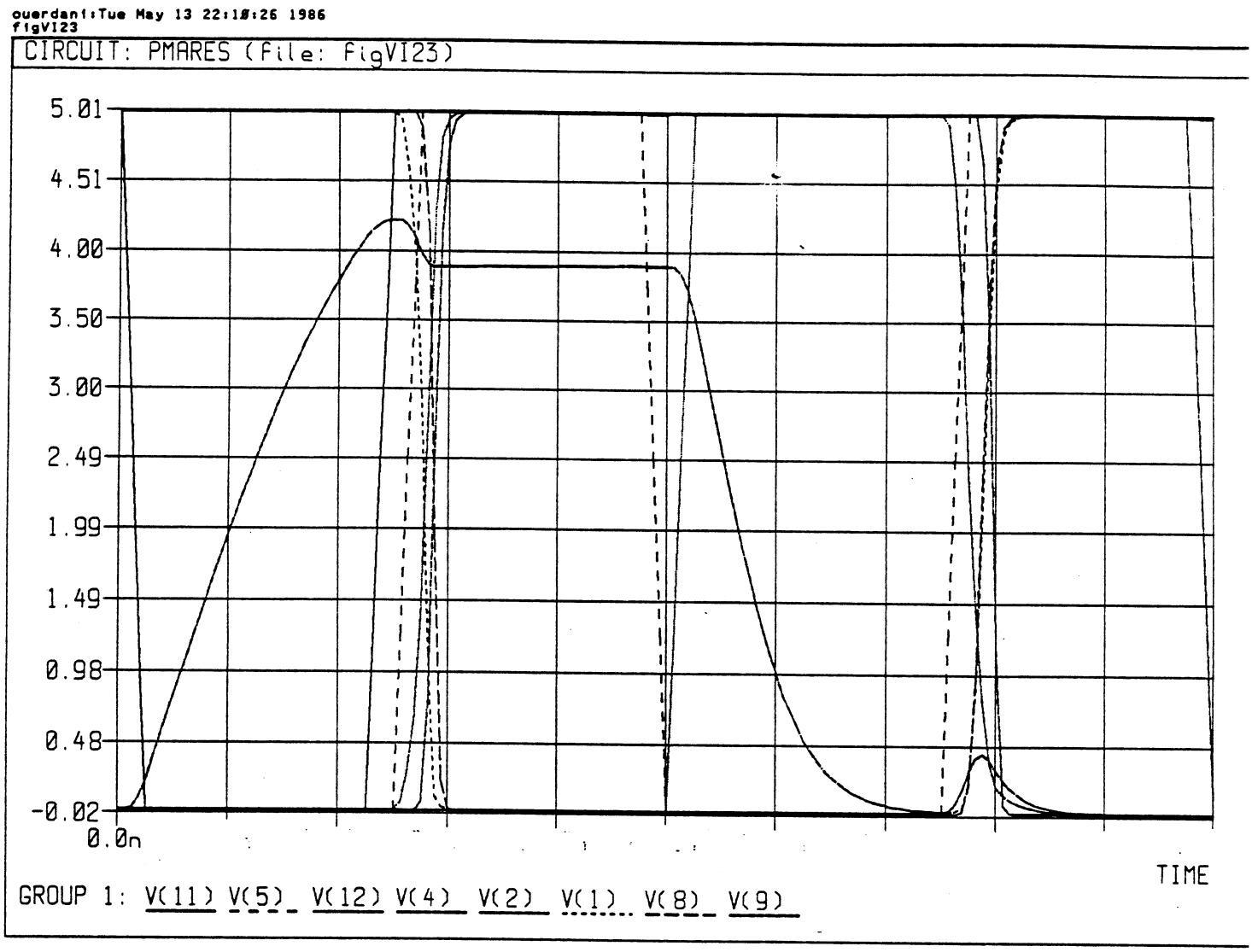


Figure VI.23: simulation de latch5

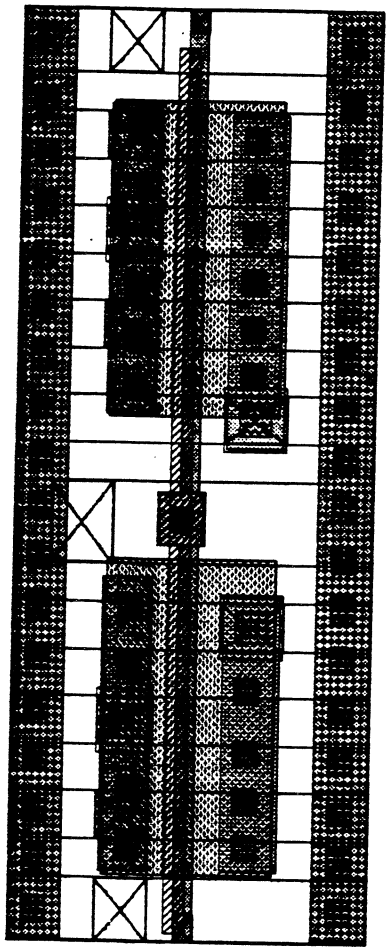


Figure VI.24: implantation des switches de passage

C O N C L U S I O N

CONCLUSION

Ce travail a porté sur la conception d'un processeur monolithique complexe. L'intérêt de concevoir un circuit remplaçant plusieures cartes déjà existantes, réside dans la comparaison de performance que l'on peut faire entre ces deux conceptions, en tenant compte du facteur technologique. Il est aussi possible d'évaluer la méthode de conception et valider les outils de CAO.

Nous avons montré comment la décomposition des instructions peut se faire par l'introduction d'une structure topologique à neuf bus séquencés par quatre phases d'horloge. L'obtention de l'algorithme d'interprétation est primordiale pour la définition de l'architecture de la Partie Opérative.

Le chapitre IV propose un algorithme de multiplication accéléré par paquet de quatre bits qui nécessite un chemin de données bien adapté. La partie Opérative obtenus possède une structure régulière de "bit-slice" (réf MEA) qui a été implantée en CMOS à deux niveaux de métalisation.

La stratégie d'implantation adoptée pour la Partie Opérative consiste en l'usage de bus de données et d'alimentations en métal2 dans le sens horizontal et une nappe de fils de commande en métal 1 dans le sens vertical. Cette représentation topologique nous a permis de faire une implantation régulière et modulaire qui réduit le coût de conception.

Cette étude a permis de préciser les besoins en outils de CAO pour faciliter la tache du concepteur, ce qui amène à faire les propositions suivantes:

1- la décomposition des instructions en un algorithme d'interprétation constitue une étape décisive dans la conception d'un processeur. Elle doit être automatisée pour optimiser les connexions des registres aux bus de la P.O,

2- il est nécessaire de construire une bibliothèque de cellules de base qui vont être appelées souvent comme les points de mémorisation.

Finalement, cette étude ne fait que confirmer l'idée que la maîtrise de la complexité des circuits futurs passe par l'utilisation de structures régulières dont la régularité est facilement exploitable par des outils de génération automatique; en d'autres termes, les VLSI de demain ne pourront plus être vus comme une suite de portes logiques mais plutôt comme un ensempble de blocs foctionnels dont le développement, l'implémentation et le test seront facilités par l'emploi d'outils spécifiques.

BIBLIOGRAPHIE

(ANC-74) ANCEAU François
"Contribution à l'étude des systèmes hiérarchisés de ressources dans l'architecture des machines informatiques."
Thèse d'état, INPG, décembre 1974.

(ANC-77) ANCEAU François
"Principes et mise en oeuvre des microprocesseurs"
Cours polycopié de l'ENSIMAG, janvier 1977

(ANC-82) ANCEAU François & REIS Ricardo
"Complex integrated circuit design strategy".
Journal of Solid State.
Circuits Vol SC-17 n°3, june 1982.

(ANC-83) ANCEAU François
"A design methodology and a silicon compiler for the VLSI circuits specified by algorithms".
Third CALTECH conference on Very Large Scale Integration.
Edited by Randal Bryant - Computer Science Press, march 1983.

(AUM) AUMIAUX M.
"Fonction logiques et arithmétiques binaire".
Logique binaire et ordinateurs.
Edition Masson & Cie.

(BAN-78) BANDARKAR D.P, JULIUSSEN J.E
"Semiconductor technologie: trends and implications".
Computer Arch. News, vol 7, No 1, Aug. 78.

(BRE-83) BREUER M.A. and KUMAR A.
"A methodology for custom VLSI layout".
IEEE Trans. on Circuits and Systems, vol. CAS-30, No.6, jun 1983, pp. 358-364.

(CAP-82) ANCEAU François & SCHOELLKOPF Jean Pierre
"CAPRI:a silicon compiler for VLSI circuits specified by algorithms".
VLSI82 Bristol University, july 1982.

(CDR-84) OUERDANI Abdelaziz
"Conceptual Design Review"
et rapport interne BULL du 22/04/84

(CHU-84) CHUQUILLANQUI Samuel
"Une nouvelle approche pour l'optimisation topologique et l'automatisation du dessin des masques de PLAs complexes".
Thèse de Docteur-Ingénieur à l'INPG, octobre 84.

(DES) ESPERS J.P. & SEQUIN H C. & VAN DE WIELE F.
"Design methodology for VLSI circuits".
Applied Science n° 47.

(DUR-79) DURET Alain
"Participation à la conception et la réalisation en LSI de la partie opérative d'une machine intégrée".
Thèse de Docteur 3ième cycle à l'INPG, décembre 1979.

(FAG-79) FAGGIN F.
"Trends in microcomputers".
Euromicro Journal, No 5, 1979.

(FOL) FOLBERTH O.G "Perspectives et limites de la micro-électronique".
01 Informatique No 17.

(GAL-84) GALLAIS R.
"Participation à la réalisation d'un microprocesseur 16 bits interprétant le P Code".
Thèse d'Ingénieur CNAM Grenoble, mars 1984.

(HEN-80) HENNION B., MAZARE G.
"CASSIOPEE, an integrated CAD system for integrated circuits".
Proc. of the ESSCIRC 80, Grenoble, sept. 1980

(HWA) HWANG Kai
"Computer Arithmetic. Principles, Architecture, and Design".
John Willey and Sons. New York

(JEC-79) JECMEN R.M and al
" A 25ns static RAM".
Proceeding of int. Solid state circuit conf. 1979.

(LAT-79) "VLSI design methodology, the problem of the 80's for microprogram design."
16th Design Automation Conference 1979.

(MAR-80) MARQUES J.M.C.A
"Mosaic: une méthodologie de conception pour les circuits systèmes VLSI".
Thèse Docteur Ingénieur, INPG, Sept. 1980

(MEA) MEAD Carver & CONWAY Lynn
"Introduction to VLSI systems".
Addison Wesley publishing company.

(MEI) MEINADIER Jean Pierre
"Structure et fonctionnement des ordinateurs"
Edition Larousse, série Informatique.

(MIC-83) LAURENT Jacques & COURTOIS Bernard
"Présentation et utilisation d'un outil de test de VLSI par faisceau d'électrons".
Rapport de recherche IMAG n° 374, mai 1983.

(REI-83) REIS Ricardo
"Evaluateur topologique prédictif pour la génération automatique des plans de masse de circuits VLSI"
Thèse de Docteur-Ingénieur, INPG juin 83.

(SA-77) Scientific American 1977

(SAU-83) SAUVEE P.
"Logique de base en technologie MOS".
Cours polycopié ENST de Paris, 1983.

(SCH-83) SCHOELLKOPF Jean Pierre
"LUBRICK: a silicon assembler and its application to datapath design for FISC".
VLSI 83, Trondheim, august 1983 (R.R. IMAG mars 1983).

(SCH-85) SCHOELLKOPF Jean Pierre
"SILICIEL: contribution à la compilation du silicium et à l'architecture des circuits intégrés"
Thèse d'Etat, INPG, avril 1985.

(SUZ-81) SUZIM Altamiro Amadeu
"Etude des parties opératives à éléments modulaires pour processeurs monolithiques"
Thèse de Docteur-Ingénieur à l'INPG, novembre 1981.

(VLS-82) ANCEAU François
"VLSI Processeur Architecture and design."
VLSI82 Bristol University, july 1982.

(WES-85) WEST N.H.E and ESHRAGHIAN Kamran
"Principle of CMOS VLSI Design. A Systems Perspective"
Addison-Wesley Publishing Compagny 1985.

A U T O R I S A T I O N de S O U T E N A N C E

VU les dispositions de l'article 15 Titre III de l'arrêté du 5 juillet 1984

VU le rapport de présentation de Monsieur F. ANCEAU

Monsieur OUERDANI Abdelaziz

est autorisé à présenter une thèse en soutenance en vue de l'obtention du titre de DOCTEUR de TROISIEME CYCLE, spécialité "Informatique".

Fait à Grenoble, le 4 juin 1986

D. BLOCH
Président
de l'Institut National Polytechnique
de Grenoble

P.O. le Vice-Président,

RESUME

La nécessité d'une conception sûre et descendante des circuits intégrés VLSI (Very Large Scale Integration) est largement reconnue. Cette thèse présente une étude sur les propriétés statiques et dynamiques des dessins de masques des principaux blocs constituant un circuit intégré réalisés en technologie CMOS. La méthode proposée est une conception par affinement successifs des spécifications. On distingue deux choix fondamentaux dans la conception des circuits:

- le choix des algorithmes,

- le choix du chemin de données associé aux blocs fonctionnels.

Les validations partielles de conception étant faites par analyse et simulation.

MOTS-CLES:

CI - VLSI - CAO - plan de masse - topologie - methodologie descendante - CMOS - assemblage de cellules - Partie Opérative - algorithme d'interprétation - niveau d'interprétation