SPÉCIFICATION ET CONCEPTION DE MICROSYSTÈMES BASÉS SUR DES CIRCUITS ASYNCHRONES

Fernando Jiménez

THÈSE

Préparée au
Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS

En vue de l'obtention du
Doctorat de l'Institut National des Sciences Appliquées de Toulouse
et de l'Université des Andes de Bogota

Spécialité :
Informatique Industrielle

par

Fernando JIMÉNEZ VARGAS
Ingénieur de l'Université des Andes

SPÉCIFICATION ET CONCEPTION DE MICROSYSTÈMES BASÉS SUR DES CIRCUITS ASYNCHRONES

ETUDE D'UN DISPOSITIF MULTICAPTEUR INTEGRÉ D'ENREGISTREMENT DE CONTRAINTES ENVIRONNEMENTALES

Soutenue le 8 Novembre 2000, devant le jury :

Directeurs de thèse	**M.**	**COURVOISIER**
	A.	**GAUTHIER**
Rapporteurs	**M.**	**STRUM**
	J.	**AGUILO**
Examinateurs	**D.**	**ESTÈVE**
	A.	**GARCIA**

Rapport LAAS N° 00596

Table des matières

Avant-propos

Les travaux présentés dans ce mémoire ont été effectués dans le groupe de recherche ORGANISATION ET CONDUITE DE SYSTEMES DISCRETS du Laboratoire d'Analyse et d'Architecture de Systèmes (LAAS) du C.N.R.S de Toulouse et dans le CENTRE DE MICROELECTRONIQUE de l'Université des Andes à Bogot, Colombie. Je remercie la Direction, l'administration et les services techniques du LAAS pour leur aide et leur attention.

Je tiens à exprimer ma reconnaissance envers les membres du jury qui par la varieté de leurs questions m'ont prouvé tout l'intérêt qu'ils ont porté à mon travail. Je remercie M. Marc COURVOISIER , professeur à l'Université Paul Sabatier de Toulouse, d'avoir accepté de présider le jury. Je remercie également Marius STRUM, professeur à l'Université de So Paulo, et Jordi AGUILO, professeur à l'Université Autonome de Barcelonne, d'avoir accepté la charge d'évaluer en qualité de raporteurs les travaux présentés.

Je tiens à exprimer toute ma gratitude envers mes directeurs de recherche pour m'avoir guidé tout au long de ces années. Je remercie Marc COURVOISIER, professeur à l'UPS, et Alain GAUTHIER, professeur à l'Université des Andes pour leur gentillese, leur enthousiasme et surtout leur amitié. Je remercie M Daniel ESTÈVE, directeur de recherche au CNRS, et Antonio GARCIA, professeur à l'Université des Andes, pour les discussion très enrichisantes à propos des problèmes lies à la conception des Microsystèmes et à la Co-Conception en Microéléctronique.

Je voudrais aussi remercier mes colegues et amis Tatiana KEMPOWSKY et Noursaid HARCHANI qui m'ont permis faire réalité le travail multidisciplinaire à distance (plus particulierement la Téléconception).

Je remercie également tous les membres du groupe OCSD et ses Directeurs M Claude HENET et Francois ROUBELLAT qui m'ont permis de passer mes séjours au LAAS dans une ambiance chaleureuse. Je remercie mes colegues du Département du Genie Eléctrique de l'Université des Andes qui m'ont soutenu pendant mes longues absences du département. A tous ceux qui m'ont aidé directement ou indirectement, merci.

Je dédie ce mémoire à mes proches, ma famille et mes amis qui ont su partager avec moi ces difficiles années de recherche.

Introduction

L'évolution poussée de l'électronique a produit une croissance des fonctionnalités ajoutées aux objets donnant lieu à des concepts tels que la mécatronique, les Micro Systèmes et l'objet intelligent. Ces nouveaux objets, qu'ils soient des machines-outils, des dispositifs au service de l'homme ou des installations à haute valeur ajoutée, présentent des caractéristiques communes:

- L'intelligence ajoutée se présente sous la forme de fonctions nouvelles telles que la surveillance, le diagnostic, la coopération, l'apprentissage, la décision, l'action et la réaction,
- Le matériel électronique (microprocesseurs, ASICs, DSPs, etc.) est embarqué, distribué, communicant, et fonctionne en temps réel,
- L'architecture du système électronique embarqué est fixée essentiellement par la structure de l'objet et de son environnement,
- Les différents sous-ensembles mécaniques, électroniques, électriques, optiques, etc. font partie intégrante du même objet, donc leur conception est liée,
- Il existe un transfert de complexité des domaines mécanique, optique et chimique aux domaines de l'électronique et du logiciel.

L'intérêt des Micro Systèmes: L'introduction des systèmes électroniques embarqués et miniaturisés dans des nouveaux dispositifs apporte des avantages qui sont ressentis essentiellement comme une réduction des coûts réels et une augmentation de la performance. Les impacts les plus importants concernent la réduction de la complexité mécanique (réduction du câblage par exemple), une meilleure opération (par l'augmentation de la flexibilité des fonctionnalités) et une maintenance plus efficace (par une meilleure connaissance de l'état de vieillissement du système).

La motivation de cette étude: La conception de tels systèmes devient de plus en plus complexe. Le résultat de l'intégration technique n'est pas accompagné par une intégration des méthodologies, des méthodes et des outils de conception. La capacité d'un individu à comprendre le détail d'une architecture complexe et multidisciplinaire se voit réduite et le besoin d'une interaction entre individus d'équipes de conception de disciplines différentes est essentiel.

Les issues: L'électronique étant maîtrisée, la compétitivité tient dans une conception, qui soit rapide, sans faute, robuste et réalisable. Les efforts doivent s'orienter vers la recherche de méthodologies et techniques qui puissent supporter l'intégration, principal défi proposé par la conception de ces nouveaux systèmes répartis, communicants et pluridisciplinaires [54].

Le Processus de Conception: La *Conception*, définie comme l'activité consistant à résoudre un problème, est la plus naturelle et la plus noble des activités humaines. Celle-ci commence avec la reconnaissance des besoins, l'insatisfaction devant l'état des choses et la réalisation que des actions doivent être entreprises pour résoudre le problème.

Nous allons étudier les paradigmes méthodologiques ou modèles de la conception de systèmes pour proposer le modèle et le processus le plus adapté a la conception de Micro Systèmes.

Une méthode de Conception peut être vue comme l'application pratique de ces paradigmes. En ce qui concerne le déroulement du processus de conception, trois stratégies sont identifiées: la *stratégie descendante*, la *stratégie ascendante* et la *stratégie mixte ascendante - descendante.*

Selon le domaine d'application, plusieurs méthodes ont été proposées, donnant lieu à une "explosion combinatoire" du nombre de méthodes de conception. Dans le cas de la *Conception de Micro Systèmes* seulement une vision globale de la conception avec un approche descendante peut permettre l'intégration recherchée.

Le processus de conception descendante d'un Micro Système quelconque se divise en deux phases séquentielles: la première (*conception amont*) comprend les activités qui vont de l'acquisition des exigences jusqu'a la génération d'une architecture validée, fiable et robuste; la deuxième (*conception aval*) dont les activités s'orientent vers la modélisation de l'intégration technologique, la simulation des conditions environnementales d'utilisation et l'étude de la physique des défaillances.

Notre contribution: Nous voulons apporter une contribution à la *conception de Micro Systèmes*, avec des points forts tels que:

- la conception amont,
- les circuits asynchrones,

- la fiabilité et,
- l'intégration au système de Conception.

La Conception amont: La *conception amont* évolue sur la base du raisonnement logique autour des informations contextuelles, des solutions possibles et de leurs applications. Le concepteur n'est pas nécessairement attentif au processus suivi, mais il est sensible à l'utilisation de toute forme de raisonnement formel pour arriver à une bonne décision de conception. Ce processus aboutit à une configuration ou architecture ayant des caractéristiques telles que:

- à toute exigence doit correspondre une sous structure du système,
- la complexité de l'interfaçage entre différentes sous structures doit rester minimale,
- le système doit être conforme aux critères d'évaluation et aux contraintes de conception fixés par la définition des exigences.

Dans les cas spécifiques qui nous concernent, le concepteur doit faire face à la partition de la solution entre différents domaines tels que mécanique, optique, électronique analogique, électronique numérique ou informatique. Une fois que l'allocation des fonctions du système aux différents domaines a été décidée, le concepteur spécialisé dans son domaine doit prendre des décisions qui peuvent affecter les décisions prises dans un autre domaine.

Les décisions à prendre dans le domaine électronique, par exemple, concernent:

- La partition entre numérique / analogique,
- Le style de conception asynchrone ou synchrone,
- l'environnement d'opération,
- les volumes de production,
- les niveaux de fiabilité,
- la facilité d'évolutions futures et l'adaptation aux changements technologiques,
- Le choix de la technologie de réalisation physique (microcontrôleurs, DSP, ASIC, FPGA, PLD, etc.).

Ces décisions conditionnent le processus en aval étant donné que chaque technologie a ses propres méthodes et outils, parfois très liés au dispositif particulier et au processus de fabrication. Cette démarche peut limiter le degré d'innovation ou le degré de modification du système complet.

Notre objectif est de proposer une méthodologie structurée [8] pour générer une architecture validée, fiable et robuste. Cette méthodologie devrait permettre le prototypage rapide d'une solution au problème posé et elle devrait se baser sur un modèle architectural avec les caractéristiques ci après:

- le modèle doit être assez puissant pour représenter à la fois les systèmes multidisciplinaires à différents niveaux d'abstraction et les différents aspects structurels, comportementaux (évolution temporelle, traitement des données, autonomie) et physiques (taille, consommation, EMI, modes de défaillances et conditions environnementales), avec un passage facile entre niveaux;

- le modèle doit être assez général pour représenter des concepts tels que la concurrence, la causalité, le choix entre événements et le parallélisme;

- le modèle doit permettre une exploration architecturale et une modélisation des performances d'une façon indépendante de la technologie de réalisation;

- le modèle doit permettre la partition fonctionnelle entre numérique / analogique et puis entre matériel / logiciel et il doit permettre le raffinement progressif d'un modèle comportemental vers un modèle réalisable au niveau RTL ("Register Transfer Level";)

- le modèle doit permettre la synthèse et la validation formelle;

- le modèle doit être basé sur des langages et des outils commerciaux;

- le modèle doit etre facile à utiliser et il doit servir comme outil de communication entre individus issus de différentes disciplines.

La démarche proposée est *descendante.* Elle vise à mettre en oeuvre une spécification exécutable de l'architecture choisie par une réalisation en langage VHDL-AMS [48] à partir de laquelle la *conception aval* peut débuter.

Nous allons nous servir de la réalisation d'un dispositif d'enregistrement de contraintes environnementales [5] pour démontrer la pertinence du modèle et

de la méthodologie proposée. Ce cas d'étude s'inscrit dans une classe de Micro Systèmes importante tant au niveau théorique que pratique. Au niveau théorique du fait de son caractère réparti, multidisciplinaire et communicant et au niveau pratique du fait de son intérêt pour la surveillance de l'état de vieillissement des installations à forte valeur ajoutée.

Nous allons essayer d'enrichir les langages industriels existants tels que VHDL et VHDL-AMS avec une sémantique formelle qui s'exprime en réseaux de Petri interprétés. Cette approche devrait permettre l'intégration et l'adoption de n'importe quel sous système basé sur cette sémantique. Ceci est très important dans la mesure où l'intérêt est porté sur la réutilisation et la composition des blocs conçus.

Les Circuits Asynchrones: Nous avons choisi les systèmes et circuits asynchrones comme *"style de conception"* étant donné les exigences des applications. Les circuits à commutation asynchrone permettent:

- la basse consommation de puissance électrique;
- la diminution de l'émission des bruits électromagnétiques.
- l'adaptation automatique aux propriétés physiques,
- le meilleur potentiel de migration technologique,
- la communication entre systèmes à différentes horloges ou sans horloge;
- la composition temporelle et fonctionnelle due à leur caractéristique d'auto temporisation;
- l'absence de retard d'horloge;
- l'augmentation des prestations grâce au paradigme du cas moyen (les circuits synchrones sont modélisés sous le paradigme du pire cas) [73];
- la spécification orientée aux interfaces, lesquelles sont asynchrones en essence.

Nous allons justifier le choix des circuits asynchrones pour l'implémentation de Micro Systèmes répartis et communicants par leur basse consommation d'énergie [64]. Nous allons chercher à combler la distance entre les modèles de haut niveau pour la modélisation des Micro Systèmes et la technologie de mise en oeuvre choisie.

La fiabilité: La classe des applications visée a des caractéristiques communes, dont la fiabilité, la robustesse et l'autonomie sont les plus importantes:

- elles sont basées sur le concept de Micro Système qui associe les fonctions capteur, actionneur, intelligence locale, communication et alimentation dans le même dispositif;
- la haute fiabilité et robustesse face aux contraintes de type environnemental et temporel;
- la sûreté de fonctionnement après de longues périodes de veille, donc la haute disponibilité;
- l'autonomie vis-à-vis de l'alimentation électrique;
- l'acquisition et le stockage des contraintes temporelles sur les quantités physiques surveillées;
- l'intégration sensorielle;
- les communications sans fil.

Nous allons proposer une architecture de surveillance supervisée à laquelle seront ajoutées des fonctions de diagnostic et de décision locale. Ceci permettra d'augmenter la fiabilité de toute l'électronique embarquée.

L'intégration au système de conception: Un système de conception global suppose:

- que l'on dispose d'une bibliothèque de composants déjà modélisés et validés;
- que l'on dispose d'une méthodologie d'assemblage et de simulation des sous ensembles et du Micro Système complet;
- que l'on dispose de composants clefs: micro capteurs, micro actionneurs pour réaliser les mesures et les commandes essentielles de gestion de l'énergie et des télécommunications,
- que l'on dispose d'un modèle de communication et d'échange entre Micro Systèmes et Micro Systèmes / centrale de décision.

Sur cette base de connaissances, nous allons mettre en place une démarche de prototypage virtuel; c'est à dire une représentation informatique en langage VHDL-AMS:

- du fonctionnement global du Micro Système,
- de l'implantation physique des composants (dimensionnement et technologie d'assemblage).

Cette méthodologie globale de Conception sera le résultat de l'intégration de la *conception amont* des architectures et de la *conception aval* concernant les micro technologies de fabrication.

Organisation de ce mémoire: Ce mémoire est organisé en cinq chapitres. Le *premier* chapitre est dédié à l'étude de la conception des systèmes microélectroniques et en particulier la Conception des Micro Systèmes. L'objectif est de montrer les besoins et les performances, les limites actuelles et ce que l'on se propose de faire. Cette étude nous permettra de proposer un modèle et une méthode adaptés à la conception des Micro Systèmes de surveillance sans fil. Le *deuxième* chapitre sera dédié à la présentation du modèle et de la méthode de conception architecturale. Nous allons montrer le rapport entre les fonctionnalités, les critères d'évaluation et la robustesse des architectures générées pour la réalisation des prototypes virtuels en VHDL-AMS. Le *troisième* chapitre sera dédié à la proposition d'une architecture électronique surveillée et son adéquation aux applications visées. Le *quatrième* chapitre concerne la réalisation d'un dispositif d'enregistrement de contraintes environnementales. La démonstration sur ce dispositif montrera l'applicabilité et la pertinence du modèle et de la méthode proposée. Finalement, dans le *cinquième* chapitre, nous allons conclure ce travail avec nos résultats et les prospectives futures de recherche dans ce domaine.

Il faut, enfin, remarquer que ce travail s'inscrit dans les Programmes de Coopération Post graduée entre la France et la Colombie. Une partie importante des résultats correspond aux transferts de technologie en conception et en micro fabrication. Nous avons cherché leur applicabilité aux problèmes colombiens notamment dans les domaines de l'environnement, de l'agroalimentaire, du transport et de la surveillance d'infrastructures face aux catastrophes naturelles. Nous proposons une plateforme de téléconception de Sytèmes Microélectroniques [1] [6] et un nouveau approche à l'education en électronique [7].

Chapitre 1

LA CONCEPTION DE SYSTÈMES ÉLECTRONIQUES EMBARQUÉS EN VUE D'UNE RÉALISATION PAR CIRCUITS ASYNCHRONES: CAS DES MICROSYSTÈMES

1.1 Introduction

Les Micro Systèmes sont des micro structures d'interfaçage entre l'environnement physique et le monde électrique. Ces systèmes ont trois caractéristiques essentielles: *la miniaturisation, la multiplicité* et *la micro électronique.* Ils sont faits de capteurs, d'actionneurs et de circuits intégrés et ils sont conçus en utili-

sant des modèles et des méthodes venus de disciplines différentes. La conception de ces systèmes est difficile par son caractère multidisciplinaire, qui lui confère sa complexité.

Par conséquent, il est nécessaire de rechercher une méthodologie structurée de conception de Micro Systèmes qui résout ces inconvénients.

Dans un environnement idéal, la conception devrait commencer par l'acquisition et la validation des exigences, besoins et contraintes du client, suivie par une description ou spécification unique donnée dans un seul langage ou modèle. La validation de cette spécification serait faite aussi à l'aide d'un formalisme unique; en suite la réalisation matérielle pourrait être entreprise. Cette dernière, demande la synthèse et l'intégration des différents composants.

Dans un environnement réel, par contre, plusieurs outils sont utilisés. La raison de cette situation est l'existence d'un modèle et d'une méthodologie pour chaque niveau d'abstraction et pour chaque discipline. De plus, ces formalismes ne sont pas assez puissants pour décrire des systèmes complexes [20]. D'autre part, les outils et méthodologies traditionnels pour "la synthèse de haut niveau de systèmes micro électroniques" présentent des difficultés spécifiques en ce qui concerne le contrôle, la prévision ou prédiction de leur comportement futur, la simulation et l'analyse des mises en oeuvre obtenues au niveau "transfert entre registres ".

Un système électronique peut avoir de multiples descriptions. Nous pouvons ainsi percevoir un espace de conception qui contient toutes les descriptions possibles d'un système, y compris la solution cherchée au problème posé. La synthèse de systèmes est le processus de raffinement d'une description abstraite d'un système, dite spécification. Ce raffinement consiste à rechercher une description équivalente mais exprimée à un niveau d'abstraction plus détaillé et proche du niveau matériel. Il est donc possible de définir la synthèse de haut niveau et la synthèse de bas niveau. La synthèse de haut niveau serait la traduction d'une description comportementale vers une description structurelle, la synthèse de bas niveau étant la traduction d'une description structurelle vers une description géométrique ou physique. Il est possible encore d'identifier plusieurs niveaux d'abstraction à l'intérieur de l'espace dans lequel un système est décrit. On peut distinguer le niveau architectural, le niveau du transfert de registres (RTL), le niveau numérique / analogique et le niveau circuit.

La description initiale est raffinée pour obtenir le meilleur choix entre les composants matériels, microélectroniques, et informatiques. Après une série de

raffinements, la synthèse du système est conclue par la réalisation des différents composants [21]. Le but premier d'un tel processus de conception de systèmes est d'assurer le succès commercial des produits conçus. Néanmoins, le succès dépend fortement de:

- l'écriture complète et approuvée des exigences et des besoins de l'utilisateur,
- la création d'une spécification du système qui soit efficace au niveau coût et qui s'adapte aux exigences fonctionnelles et non-fonctionnelles,
- la réalisation d'une implémentation qui soit conforme à cette conception,
- le développement dans les délais et budgets prévus, et
- la facilité avec laquelle les personnes chargées du maintien et de l'évolution du produit comprendront sa conception et sa réalisation.

Par conséquent, le succès d'un produit demande une démarche qui tient aux critères suivants [22]:

- avoir une méthodologie organisée pour la spécification, l'analyse et la conception d'un système digital et
- avoir une bonne documentation tout au long du processus, de la saisie des spécifications jusqu'à la production.

Nous voulons apporter une contribution à la *conception* de Micro Systèmes en proposant une méthodologie de Conception *descendante*, une méthode d'implémentation de cette méthodologie et un modèle formel ayant comme caractéristiques les éléments suivants :

- La méthodologie doit permettre l'intégration de la conception de haut niveau, de la synthèse architecturale et de la conception physique des Micro Systèmes ;
- La méthodologie doit fournir un formalisme unique pour acquérir les exigences, les besoins et les contraintes ;
- Ce formalisme doit permettre la validation des exigences, des besoins et des contraintes et il doit produire une spécification unique et globale du Micro Système ;

- Cette spécification doit être exécutable pour valider et vérifier les spécifications vis à vis des exigences;

- Cette spécification forme la première description fonctionnelle du Micro Système à concevoir a partir de laquelle tous les raffinements ultérieurs doivent être validés;

- Cette spécification exécutable doit permettre l'analyse des propriétés d'une solution au problème de conception et elle doit être le lien direct vers les outils d'implémentation;

- La méthodologie doit donner les guides pour le développement des modèles pour les différents composants du Micro Système et indiquer comment les intégrer pour réaliser un prototypage virtuel ou simulation.

- La méthodologie doit être hiérarchique avec des niveaux d'abstraction assez élevés pour pouvoir représenter le monde des Micro Systèmes (le monde est asynchrone, continu et chaotique).

Dans la suite de ce chapitre, nous allons étudier les propriétés et la structure du processus de conception et les méthodes utilisées pour l'acquisition des exigences, des besoins et des contraintes. Nous allons nous pencher sur la description comportementale et architecturale des systèmes concurrents et fonctionnant en temps réel. Ces méthodes font appel à des modèles formels pour la spécification et la validation des concepts de fonctionnement et pour la vérification des contraintes. Ce choix dépend en grande partie du domaine d'application et du niveau de complexité. Nous allons essayer de trouver les mieux adaptés à notre domaine d'application. Le recours à une diversité de modèles provient de la nécessité de représenter les différents aspects d'un système: fonctionnel, informationnel et événementiel. La tendance est de proposer un modèle par aspect. *La cohérence de la méthode repose sur le fait que ces divers modèles s'articulent et s'intègrent de façon à représenter tout le système sans omission, ni répétition* (voir figure 1.1).

Nous conclurons enfin ce chapitre quant à l'adéquation de ces résultats par rapport au problème de la conception et de la représentation des Micro Systèmes.

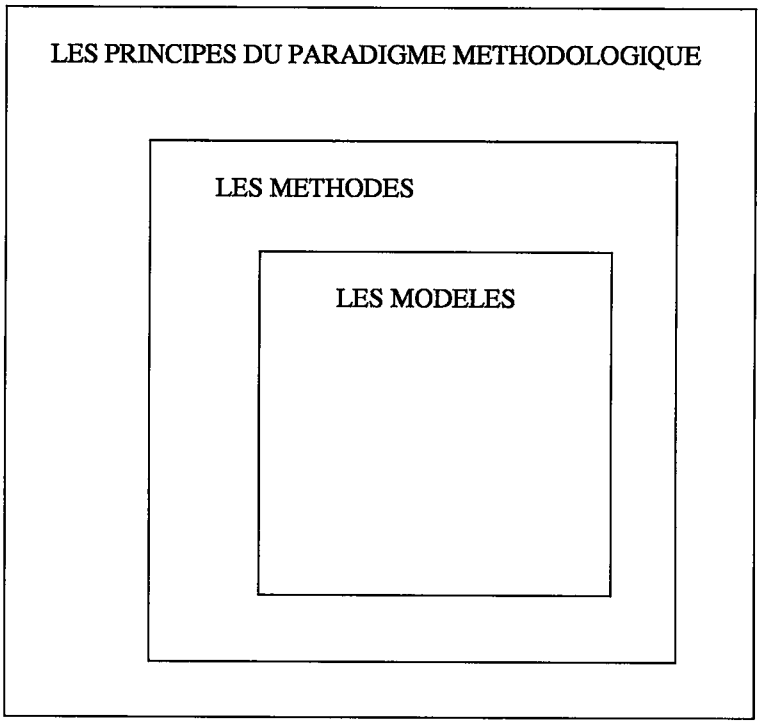

FIG. 1.1 – *Une méthodologie de Conception : les principes, les méthodes et les modèles*

1.2 Les Performances Actuelles et les limites de la Conception des Systèmes Electroniques

L'état actuel de la Conception des systèmes micro électroniques, demande des nouvelles approches au processus de conception et des changements profonds à la méthodologie de conception. La productivité de la conception doit être augmentée, si l'on veut profiter des avancées technologiques et si l'on ne veut pas compromettre les grands investissements faits pour le développement des technologies de fabrication. Une amélioration sur une partie du processus peut avoir un impact important sur les coûts et la productivité.

La gestion de la complexité est à la source des limites actuelles de la conception des systèmes électroniques.

Cette complexité peut être vue sous différents aspects :

Le statut et l'évolution de la technologie [11]: L'effet de l'évolution technologique a deux facettes importantes : la complexité des systèmes électroniques et les nouveaux paradigmes pour la conception des circuits intégrés.

La complexité des systèmes électroniques est définie comme le nombre de transistors par circuit intégré. Elle s'accroît en moyenne de 60% par an du fait des progrès de la technologie des procédés de fabrication. Par contre, la productivité des concepteurs ne s'accroît que de 20% par an produisant un retard de la conception par rapport à la technologie. Ceci entraîne une demande de concepteurs de plus en plus importante. La taille des équipes de conception atteint ses limites, tant en ce qui concerne les interactions qu'en ce qui concerne le retour de l'investissement.

Le paradigme dominant pour la conception des circuits VLSI à été la logique statique CMOS, mais la tendance croissante vers les "System on a Chip" [72] (effets au niveau sub-micron) et les Micro Systèmes et l'explosion du marché des produits portables et communicants oblige à ce que les systèmes conçus ajoutent une partie analogique, RF, électromécanique, électrothermique, optique et aussi chimique pour s'interfacer avec le monde externe.

Les méthodologies et les outils CAO : Les méthodologies existantes limitent l'intégration du niveau système parce que le concepteur ne peut pas passer facilement entre les niveaux d'abstraction, des détails du dispositif jusqu'à la fonctionnalité du produit. Un panorama des outils CAO pour les systèmes

électroniques reflète les problèmes suivants: les manques de standards et d'interopérabilité compliquent le passage au travers des niveaux d'abstraction; la spécification des objectifs de la conception sont en général informels; les paramètres de l'environnement et de la performance du système se transmettent sans l'utilisation d'un modèle comportemental et temporel bien défini; la synthèse des architectures conçues se fait avec un approche *ascendante* en utilisant un mélange de méthodes manuelles, des outils de synthèse logique et de génération des masques géométriques semi automatiques; la vérification se fait par simulation à differents niveaux d'abstraction avec des résultats déconnectés; la testabilité n'est pas expressément incorporée à la conception; les techniques d'auto test incorporé et de balayage par la frontière ne sont pas encore communs sur les produits du marché.

Le cas des Micro Systèmes: Aujourd'hui, les Micro Systèmes sont conçus en utilisant souvent l'analyse par éléments finis ou l'analyse par flot de données. Les méthodologies actuelles ont les défauts de ces deux outils: les méthodes basées sur l'analyse par éléments finis sont très lourdes en temps de conception et en temps de calcul, néanmoins elles restent assez générales. La modélisation se fait à un niveau d'abstraction proche du niveau physique en on ne dispose pas de hiérarchie. Par ailleurs, l'analyse par flots de signaux offre un niveau d'abstraction assez élevé mais il n'existe pas de liaison entre le modèle comportemental et le modèle physique. Aucun des deux n'intègre la conception des sous systèmes micro électroniques. La conception de Micro Systèmes exige aussi la modélisation des défaillances analogiques qui perturbent le fonctionnement effectif du Micro Système. Les défaillances typiques peuvent se classer en défaillances paramétriques et défaillances catastrophiques. Les défaillances paramétriques sont dues à des déviations des tolérances des paramètres physiques de fabrication, et les défaillances catastrophiques sont dues à des changements drastiques pendant le micro usinage ou pendant la vie utile du Micro Système.

1.3 Un Paradigme pour la Conception de Micro Systèmes

La Conception étant une activité humaine naturelle, elle est omniprésente et pourtant difficile à définir. Derrière les activités de conception, il existe des

principes, des règles, des lois et des techniques ; néanmoins il n'existe pas un ensemble standard de règles de conception applicables à toutes les classes de problèmes de Conception et elle reste toujours une activité partiellement subjective.

Braha *et al* ont identifié les caractéristiques fondamentales de la conception et ont examiné sa logique et sa méthodologie en ingénierie. Le lecteur peut trouver une étude approfondie dans [12].

Dans les cas de la Conception des Micro Systèmes ces caractéristiques peuvent être énoncées comme suit :

- La prise de décisions pendant les activités de conception est très complexe étant donné les différents domaines concernés (électrostatique, mécanique, thermique, chimique, optique, microélectronique, etc.) ;
- Les options, les alternatives et les résultats attendus ne sont pas connus d'avance et une activité de recherche importante doit être entreprise (l'intégration technologique, l'alignement des microstructures, l'amélioration de la fiabilité etc.) ;
- Souvent les décisions optimales ne sont pas trouvées et une solution satisfaisante sera acceptée (modélisation comportementale contre modélisation physique, simulation par éléments finis, prise en compte de l'environnement, modélisation des modes de défaillance, analyse des tolérances, etc.).

Ce caractère est accentué dans le cas des Micro Systèmes notamment par :

- l'évolution rapide et sans cesse des micro technologies,
- la pression sur le concepteur pour résoudre le problème de conception dans des délais de plus en plus courts,
- le besoin de rechercher des nouveaux composants ou micro structures innovants.

Le caractère évolutif de la conception peut être mis à profit dans les cas des Micro Systèmes par les activités suivantes :

- Adapter des Micro Systèmes déjà conçus pour être utilisés dans une nouvelle application si son environnement interne s'adapte à son environnement externe et vice-versa, donnant lieu à des familles de Micro Systèmes ;

- Décomposer leur processus de conception en cycles de conception plus petits de manière à les appliquer récursivement pour trouver une solution satisfaisante ;
- Permettre activement la nouvelle conception des dispositifs existants mettant en cause les concepts fixes tout en essayant d'innover au niveau des sous systèmes et des composants.

Dans ce contexte, un paradigme de conception de Micro Systèmes doit inclure deux concepts relationnels :

- une matrice multidisciplinaire de théories et de techniques,
- un ensemble de cas d'étude issus de la recherche scientifique ou de la veille technologique.

Ce paradigme doit se matérialiser par un modèle abstrait préscriptif du processus de conception qui soit le point de départ pour résoudre un problème de conception de Micro Systèmes. Ce modèle ou paradigme doit servir de cadre pour bâtir des méthodes de conception pratiques, procédures et outils informatisés pour automatiser le processus de conception.

Nous allons adopter un paradigme de conception avec les caractéristiques suivantes :

- *Un composant est décrit en termes de ses parties à n'importe quel niveau d'abstraction. Chaque partie est décrite par ses attributs. Chaque attribut est décrit par ses dimensions ;*
- *Les spécifications ou les contraintes sont les caractéristiques fonctionnelles, comportementales, de performance, de fiabilité, et autres qui doivent être présentes dans l'implémentation physique du composant. Une contrainte de haut niveau est une propriété qui doit être satisfaite par les contraintes de niveau inférieur ;*
- *La conception évolue par cycles successifs. Le mécanisme de l'évolution est une activité de vérification de la vraisemblance des spécifications existantes et d'introduction des nouvelles spécifications et des nouveaux paramètres.*

Illustrons ces caractéristiques par deux exemples :

- La contrainte A("assurer une basse consommation de puissance") est considérée comme validée si et seulement si la contrainte B("le système a un

mode de fonctionnement en veille quand il n'y a pas d'activité") est valide. A son tour B est considéré validé si et seulement si la contrainte C("tous les blocs fonctionnels du système peuvent être actifs ou inactifs selon le mode de fonctionnement ") est satisfaite. Or le bloc mémoire ne peut jamais être éteint sur danger de perte de l'information, donc le type de bloc mémoire doit être non volatile, et un nouveau paramètre se génère: "le type de volatilité des blocs mémoire". L' *outil* utilisé pour supporter la contrainte C doit vérifier que tous les blocs (différents des blocs mémoire) peuvent être allumés ou éteints et que les blocs mémoire sont du type non volatile.

- La vraisemblance de la contrainte D("sélectionner une antenne insensible aux interférences de proximité")est déterminée par le processus de sélection du type d'antenne, lequel doit évaluer le degré d'efficience de chaque alternative par rapport à des critères bien définis. Ainsi, la contrainte est validée si le paramètre ("l'antenne est de type boucle") a été sélectionné.

Les dépendances entre spécifications et paramètres sont représentées par des relations logiques ou règles. La vraisemblance d'une règle implique la vraisemblance de la spécification. Par exemple, vraisemblance de la spécification A("l'antenne doit être intégrée au Micro Système") est déterminée par la relation logique ("la surface minimale de l'antenne (SA) doit être spécifiée" et "la surface dédiée à l'antenne dans le Micro Système (SM) doit être déterminée" et " SA doit être plus petit que SM"). Ainsi, la spécification A est remplacée par la règle correspondante et la spécification ("la surface minimale de l'antenne (SA) doit être spécifiée") génère un nouveau paramètre ("l'aire interne de la boucle de l'antenne doit être entre 4 et 10 cm2").

La caractéristique distinctive de ce modèle est son caractère évolutif: le processus de conception est *constamment sujet à des révisions.* Dans notre exemple, si la spécification A n'est pas satisfaite, le concepteur doit changer le type d'antenne, ou la taille du Micro Système ou remettre en question les performances demandées au système d'émission / réception RF.

1.4 Les Méthodologies et les Méthodes

La méthodologie de Conception est l'entreprise commune aux communautés des concepteurs et des fabricants des outils de CAO. Une méthode est l'implé-

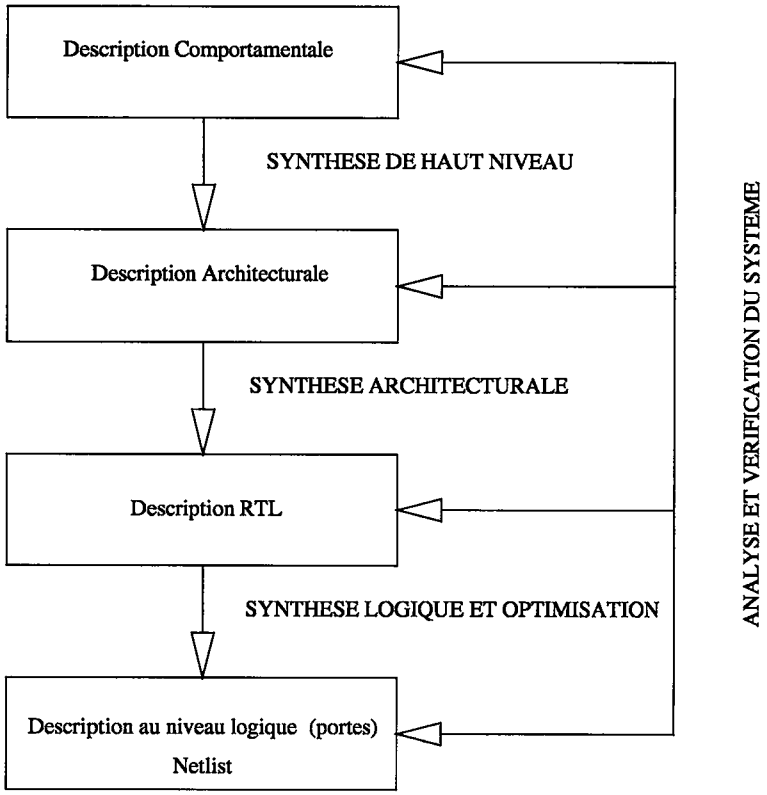

FIG. 1.2 – *Conception assistée par ordinateur de Systèmes Électroniques*

mentation pratique des étapes d'une méthodologie.

La figure 1.2 montre les différentes étapes pendant le processus de conception et d'implémentation d'un système électronique (ESDA-Electronic System Design Automation).

La Conception se divise en deux niveaux: au plus haut niveau se trouve la Conception Système (*conception amont*) où une architecture doit réaliser les fonctionnalités demandées au produit final; et au plus bas niveau se trouvent les blocs primitifs (portes logiques, amplificateurs, micro capteurs, etc.) dépendant fortement de la technologie sous-jacente (*conception aval*). Ainsi, la Conception globale peut se définir comme la création d'architectures complexes en utilisant des blocs primitifs à l'aide des outils de CAO. Par conséquent, la performance du système est fortement fonction des blocs primitifs, dont les performances et fonctionnalités atteignent leur limites, ainsi que la complexité des applications qui voit une croissance super exponentielle.

Les règles traditionnelles de la conception sont remises en question par ces tendances. Les règles plus amplement utilisées aujourd'hui sont:

- La synchronisation globale du système;
- L'utilisation du transistor CMOS comme dispositif de commutation dont la taille est réduite pour retrouver des performances accrues (introduisant une variabilité statistique du comportement du transistor);
- La capacité de produire des "puces" à un taux de 0 % défauts.

Nous allons nous centrer, en première instance, sur la conception asynchrone du niveau système en nous appuyant sur:

- *Des méthodes pour l'acquisition des exigences, des besoins et des contraintes et pour leur conversion en spécifications et par la suite en modèles exécutables (écrits en langages de description et de spécification de haut niveau);*
- *Des méthodes pour la synthèse "hardware" a partir des spécifications du niveau système;*

Les méthodes d'acquisition des exigences et de spécification des systèmes temps réel appartiennent à deux types d'approches: des approches fonctionnelles et des approches orientées objet. Le lecteur peut trouver un étude approfondie dans [13].

1.4.1 Méthodes Fonctionnelles

Les méthodes fonctionnelles sont basées sur une décomposition du système suivant les fonctions qu'il doit assurer. Elles sont composées:

- de modèles et de règles d'assemblage, constituant un langage de représentation,
- d'une démarche de modélisation qui permet de construire une ou plusieurs spécifications du système.

Les méthodes les plus adaptées à nos applications sont: SADT et SA-RT.

La méthode " Structured Analysis and Design Technique " (SADT par SOFTECH 1977)

Cette méthode couvre essentiellement la phase d'analyse des besoins ; elle permet de bien cerner les problèmes posés en ce qui concerne un cahier des charges avant de chercher à en exposer une solution.

Le modèle sous-jacent est constitué d'un ensemble de diagrammes ordonnés hiérarchiquement. Ces diagrammes sont construits d'une façon descendante à l'aide de " blocs " liés qui correspondent à des activités ou des données selon l'aspect concerné par la description. Chacun de ces diagrammes peut être considéré soit comme un diagramme père, synthèse de ce diagramme fils, soit comme un diagramme fils qui constitue un affinement d'un " bloc " de son diagramme père. Le modèle obtenu est un modèle hiérarchique avec plusieurs niveaux d'abstraction, le premier niveau décrit à l'aide d'un seul " bloc " les interactions du système à analyser avec son environnement.

Cette méthode est considérée comme générale et applicable à l'analyse de tout système. Elle est utilisée essentiellement en amont de toute méthode de spécification et de conception tout en facilitant le travail d'équipes pour le développement de grands projets. Cependant, elle ne permet pas un passage direct à la phase de conception. L'inconvénient majeur de cette méthode réside dans la pauvreté de l'expression du contrôle. Les modèles obtenus sont des modèles statiques qui ne peuvent pas être implémentés.

La méthode "Structured Analysis for Real Time " (SA-RT par Ward-Mellor 1985 et Hartley-Pirbhai 1987)

SA-RT est une méthodologie globale couvrant les différentes phases de développement. L'analyse structurée temps réel propose des extensions temps réel de l'analyse structurée de DeMarco. Ces extensions portent sur l'adjonction des fonctionnalités de contrôle décrivant la dynamique du système et le traitement des données correspondantes (les signaux événementiels).

Cette méthode est basée sur l'élaboration de deux modèles, un modèle logique de spécification d'un système, appelé modèle essentiel (Ward-Mellor) ou modèle des besoins (Hateley-Pirbhai), et un modèle technologique ou de conception désigné par modèle d'implémentation (Ward-Mellor) ou modèle d'architecture (Hateley-Pirbhai).

Le modèle technologique, issu de la conception, porte sur l'architecture d'un

système. C'est un début de solution, à la fois matérielle et logicielle, liée au choix d'une technologie particulière, et apte à supporter les fonctionnalités, à résoudre les contraintes, et à atteindre les performances spécifiées.

Le modèle logique, issu de la spécification concerne les aspects fonctionnels, informationnels et événementiels d'un système temps réel. L'aspect fonctionnel porte sur les fonctionnalités que le système doit présenter et l'aspect informationnel concerne les données mises en jeu pour leur accomplissement. L'aspect événementiel est spécifique des systèmes temps réel, qui doivent répondre à des signaux en provenance de l'environnement, et réaliser des transformations de données conditionnées par des contrôles.

La méthode fournit une approche structurée qui donne le moyen au concepteur de mener une spécification modulaire et hiérarchisée de façon à maîtriser la complexité des systèmes à concevoir. Cette approche propose des outils qui couvrent les différentes vues d'un système: fonctionnelle, événementielle et informationnelle.

Cependant, nous pouvons constater une limitation essentielle de cette méthode pour le domaine d'application qui nous concerne: Les machines à états finis s'adaptent mal à la spécification et à la conception des systèmes complexes asynchrones et grand degré de parallélisme et cela à cause de l'explosion rapide du nombre d'états et au manque de hiérarchisation des signaux. Par ailleurs les réalisations produites (synchrones) sont à haute consommation de puissance électrique et à haute émission de bruits électromagnétiques (haute susceptibilité). Il manque aussi le lien entre cette méthode et les phases de synthèses architecturales et logiques de systèmes électroniques.

1.4.2 Méthodes Orientées Objet

Par opposition aux méthodes classiques de décomposition, les approches orientées objet regroupent (encapsulation)[49], dans le concept objet, à la fois des données et des opérations sur ces données. Un objet inclut aussi une description du comportement interne qui spécifie l'ordre de déroulement des opérations internes à l'objet et la façon dont l'objet réagit aux événements qui se produisent.

Chaque objet présente à son environnement une interface bien définie dans laquelle il offre les opérations (ou méthodes) et les données partagées. Toute communication avec l'objet doit passer par cette interface.

La différence entre l'analyse orientée objet et l'analyse fonctionnelle réside dans les moyens d'expression. L'analyse fonctionnelle étudie les fonctionnalités demandées au système séparément de ses données, tandis que l'analyse orientée objet utilise des concepts qui facilitent grandement les créations et mises à jour d'objets.

Dans le domaine du développement d'applications temps réel, trois méthodes sont exploitables d'une façon industrielle. Il s'agit de la méthode OOA/OOD de Booch, de la méthode HOOD développée sur l'initiative de l'Agence Spatiale Européenne et de la méthode OMT de Rumbaugh et al.

Les méthodes orientées objet offrent des concepts qui permettent une modélisation assez générale applicable à tout type de système et à un très haut niveau d'abstraction. Ces méthodes sont plutôt adaptées à l'analyse et à la conception des systèmes logiciels temps réel et peu de travaux ont été faits pour étendre ces concepts à la conception des systèmes électroniques, notamment en ce qui concerne la co-conception.

Toutes les méthodologies orientées objet souffrent d'un manque de formalisation de la phase d'élaboration du modèle des objets.

Par ailleurs, il n'existe pas des liens entre ces méthodes et la synthèse architecturale et logique de systèmes électroniques.

Nous allons retenir la méthode SA-RT et nous allons lui emprunter les concepts qui concernent la spécification et description hiérarchisées et la partition entre traitement des données et son contrôle pour avoir à spécifier dès le début d'une manière spécifique tous les aspects de commande des architectures générées. Ainsi, nous pouvons assurer d'avoir une spécification système unique qui comprend la conception dans sa totalité et en référence à laquelle tous les raffinements successifs vont être confrontés. Cette méthode nous assure aussi la simplicité et l'adéquation aux applications envisagées. Ce choix va nous permettre, aussi, de définir un environnement unique pour la synthèse "hardware" à partir des spécifications du niveau système. Des méthodes orientées objet nous allons retenir le concept d'encapsulation du traitement des données et de son contrôle (comportement) en objets que nous allons appeler composants. *Ces composants offriront à son environnement une interface bien définie que nous allons appeler* les services du composant *de manière a favoriser la modularité et la création des architectures orientées aux connexions. Ainsi, le concepteur devra dès le début de la conception bien définir les services que l'un des composants offre ou demande aux autres composants de l'architecture. En même temps*

il ou elle pourra ré-utiliser un composant déjà crée et dont son comportement a été bien vérifié et testé en se rassurant, uniquement, que les services offerts et demandés sont bien compatibles avec les composants avec lesquels il interagisse.

1.5 Les Modèles et les Langages

L'industrie des semi-conducteurs et des Micro Systèmes demande des modèles mathématiques précis des systèmes et des architectures, pour pouvoir garantir la conformité du modèle avec des modèles des niveaux d'abstraction inférieurs du même système en conception. Nous allons proposer un approche multi modèles étant donné les exigences de nos applications (multidisciplinaires, asynchrones, concurrents et fonctionnant en temps réel). Cette approche devrait permettre une vérification structurelle et une validation formelle à très haut niveau et l'implémentation d'une liaison vers les outils de génération des blocs, de synthèse logique et de synthèse comportementale. Les modèles et langages que nous allons utiliser seront présentés par la suite.

1.5.1 Les modèles et les langages pour les Systèmes à Evénements Discrets

Les réseaux de Petri Interpretés

Les règles d'évolution des réseaux de Petri mettent en évidence des relations de cause à effet du système à événements discrets modélisé. Les places d'entrée et l'interprétation de la transition sont les conditions de l'évolution modélisée par la transition, tandis que les actions et les places de sortie en montrent explicitement les conséquences.

Ils sont très utilisés dans le domaine des systèmes à événements discrets parce qu'ils permettent une modélisation facile du parallélisme (plusieurs jetons dans le même réseau), de la synchronisation (transition avec plusieurs arcs d'entrée) et de l'alternative (place avec plus d'une transition de sortie),.

Une autre facette des réseaux de Petri est la possibilité de faire la preuve formelle de certaines propriétés dites " bonnes propriétés " : Réseau vivant, Réseau borné, Réseau ré-initialisable.

Des méthodes d'analyse par réduction du graphe, arbre des marquages accessibles ou recherche d'invariants permettent l'analyse des réseaux de Petri.

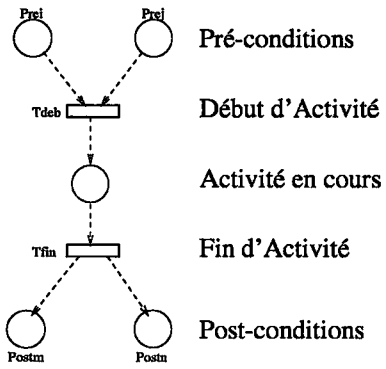

FIG. 1.3 – *Activité de Commande modélisée par Réseaux de Petri Interpreté*

Elles autorisent donc une forme de validation des spécifications établies à l'aide de cet outil.

Nous allons retenir le concept d'activité de commande modélisé par des réseaux de Petri interprétés.

Une activité de commande est délimitée par deux événements représentés par deux transitions tdeb et tfin du réseau de Petri. Chacune de ces transitions conditionne la validité de l'événement par des tests correspondant à des conditions de tir. Ces tests concernent les places d'entrées Prei et Prej de tdeb appelées pré-conditions. Lorsqu'une activité de commande est terminée, la transition tfin est sensibilisée. Son tir entraîne le marquage des places Postm et Postn en général appelées post-conditions (voir figure 1.3).

Dans le cas des systèmes asynchrones, les principes de base des réseaux de Petri sont naturels pour la description de tels systèmes. Le choix des réseaux de Petri comme modèle de base est justifié parce que:

- les réseaux de Petri sont un formalisme naturel pour décrire les relations cause - effet, la concurrence des événements et les conditions de conflit;
- les réseaux de Petri permettent une vérification et une analyse formelles assez simples et de base;
- le coût actuel de la puissance de calcul permet la faisabilité d'opérations intensives basées sur les réseaux de Petri;
- les réseaux de Petri permettent de combler la distance entre la description comportementale et la description basée en machines à états finis

Le Langage de Description des Spécifications SDL

SDL (Specification Description Language) est un langage opérationnel défini en 1982 par l'ISO pour la spécification de systèmes ouverts et par la CCITT en 1988 pour la spécification de systèmes de télécommunications et de réseaux numériques. SDL fournit une représentation graphique et une représentation textuelle. La représentation du système à spécifier est donnée en termes de blocs structuraux et des canaux de communication. Le langage permet l'implémentation des méthodologies descendantes par sa caractéristique de raffinement structurale. Un bloc peut être spécifié comme un ensemble de sous blocs ou un ensemble de processus. Le comportement d'un bloc est spécifié à l'aide des machines à état étendues par des diagrammes de flot de données. Ces machines montrent les états, les transitions et les sélections (IF statements). Les événements sur les entrées et les sorties sont associées aux processus. Seules les entrées sont associées aux transitions et elles représentent des conditions de changement d'état. En SDL, le temps est modélisé par l'intermédiaire de temporisateurs. Un temporisateur est un processus indépendant qui peut recevoir et/ou envoyer des messages.

SDL est le langage le plus adapté à notre domaine d'application étant donné que c'est un langage standard pour la spécification et description de systèmes distribués. La faiblesse de cette notation est l'utilisation des machines à états pour décrire le comportement d'une sous structure. Les machines à états finies s'adaptent mal à la spécification et à la conception des systèmes complexes asynchrones et à grand degré de parallélisme et cela à cause de l'explosion rapide du nombre des états, au manque de hiérarchisation des signaux, à la haute consommation de puissance électrique et à la haute émission de bruits électromagnétiques (haute susceptibilité) des implémentations matérielles. Par contre, les méthodologies bâties au tour de SDL fournissent des liens vers les phases de synthèse architecturale et logique de systèmes électroniques, surtout en ce qui concerne la co-conception. Nous allons étendre le langage SDL en proposant les réseaux de Petri Interprétés comme modèle pour décrire le comportement des blocs structurels. Ceci nous permettra de soulever les problèmes issus de l'utilisation des machines à états et amener une description asynchrone aux niveaux de description élémentaire de SDL.

1.5.2 Les Modèles et les Langages pour les Systèmes Analogiques

Les modèles utilisés pour la modélisation analogique représentent les lois définissant l'interaction entre les composants du système par des équations à temps continu. Ces lois peuvent appartenir aux domaines physiques (mécanique, état solide, optique, chimique, etc.) ou aux domaines non physiques tels que ceux de l'économie, de la technologie, etc. Les modèles sont construits par des blocs qui interagissent par des signaux à temps continu.

Il existe deux approches de modélisation en temps continu:

- La Modélisation Conservatoire décrit un système par un ensemble de composants, chaque composant étant décrit par une relation entre deux variables dont le produit se conserve (puissance, énergie) ; cette relation se modélise par un graphe, dont les sommets représentent les noeuds équipotentiels du système et les branches représentent les éléments au travers des quels le flux d'énergie se transforme ou se stocke. Les variables inconnues décrivant un tel système son les *Quantités de branche.* Il existe deux types de quantités de branche : la *quantité au travers* (courent, force, débit, flux de chaleur, etc.) et la *quantité entre* terminaux de la branche ou du graphe (tension, vitesse, pression, température, etc.). Les deux terminaux d'une branche ou d'un port d'un graphe sont déclarées comme étant d'une certaine nature physique (électrique, mécanique, fluidique, thermique).

- La Modélisation par Flux de Signaux décrit un système par ses composants. Chaque composant ou entité est spécifié par une relation algébrique ou différentielle entre les signaux d'entrée et ceux de sortie.

Cette modélisation présente des avantages tels que :

- La modélisation peut se faire à des niveaux d'abstraction très élevés,

- La modélisation au niveau système est très flexible ;

- Les modèles sont compatibles avec les lois de conservation de la matière et de l'énergie.

- Les modèles servent à faire des descriptions très proches du niveau physique.

La solution au modèle correspond au point ou points de l'espace d'état où toutes les fonctions de temps continu satisfont toutes les relations. La solution est calculée en utilisant des algorithmes pour résoudre des systèmes d'équations différentielles.

Le temps est une variable réelle qui évolue indépendamment, mais qui est discrétisé par les algorithmes d'intégration. Ceci donne lieu à des erreurs d'intégration, à des discontinuités et à des problèmes de convergence des algorithmes de calcul. Néanmoins, le temps est global et tous les composants partagent las même notion de temps. Les langages les plus utilisés pour la modélisation analogique sont :

- Matlab-Simulink

- Spice

La modélisation analogique permet de simuler un système de la manière la plus précise et la plus proche du comportement réel. C'est par la modélisation analogique que l'on descend les niveaux d'abstraction du comportement fonctionnel au comportement physique ; par exemple, la modélisation de l'environnement d'opération, la modélisation de l'injection de fautes, l'étude du comportement dynamique des micro et des nano-structures, etc. Les faiblesses de cette méthode sont le manque de hiérarchie et le manque de lien entre la conception et la spécification des systèmes électroniques.

1.5.3 Les Modèles et les Langages de Simulation Hybride

La modélisation Hybride est constituée par les modèles hétérogènes qui combinent au moins deux des formalismes présentés auparavant. Cette tendance est accentuée par la nécessité de faire des prototypes plus complets et plus proches de la réalité. Cette approche correspond à l'utilisation d'un modèle différent pour différentes vues du même système. Ainsi, SA-RT/PNO combine l'analyse structurée avec les réseaux de Petri à Objets. Les réseaux de Petri Hybrides représentent la cohabitation du monde discret avec celui du continu. Pour nos applications, il est nécessaire de pouvoir donner au concepteur l'option de modéliser un comportement comme un système à événements discrets ou comme un système à temps continu selon le niveau d'abstraction souhaité. Ces deux comportements doivent pouvoir coexister à l'intérieur de la même description.

Cette tendance a été reconnue par la norme IEEE 106.1 VHDL-AMS qui fournit un cadre général pour l'interaction entre ces deux modèles.

La modélisation Hybride a été suivie et appuyée par la nouvelle génération de langages de simulation tels que Verias HDL, Verilog AHDL ou Anacad de Mentor, compatibles avec la norme VHDL-AMS, fournissant un environnement de simulation hybride discret - continu tout en gardant les avantages du langage VHDL traditionnel.

1.6 Les particularités des Systèmes et des Circuits Asynchrones

Les modèles destinés à la Conception des systèmes et des circuits asynchrones peuvent être classés selon le point de vue du système en trois catégories: modèles orientés flot de données ou architecturaux, modèles orientés flot de contrôle ou comportementaux et modèles mixtes présentant les deux orientations données et contrôle.

Les modèles et les outils proposés jusqu'à présent peuvent être comparés selon les caractéristiques suivantes:

- Primitives de calcul
- Primitives de spécification (concurrence, hiérarchie comportementale, hiérarchie structurelle, expression du choix, transitions d'état, exceptions, etc.)
- Principes opérationnels des circuits asynchrones (prise en compte des retards, prise en compte des aléas, protocole de signal, etc.)
- Méthode de synthèse,
- Applications.

Les modèles que nous avons étudiés sont:

NOM DU MODÈLE	AUTEUR ET DATE
FSM asynchrone	HUFFMAN 1969, UNGER 1971
Machines auto-synchronisées	UNGER 1977, AGHDASI 1991, LADD 1991
Machine à données	DAVIS 1979
FSM avec horloge local	NOWICK et DILL 1993
Réseaux de Petri et graphe des données	COURVOISIER et VALETTE 1980
Machine en mode rafale étendu	YUN, DILL et NOWICK 1991
Asynchronous Logic Arrays	PATIL 1975
Graphes de signal	YAKOLEV 1985
I nets	MOLNAR 1985
Signal Transition Graphs STG	CHU 1992, CORTADELLA et al. 1994
State Graphs	BEEREL 1992, KONDRATYEV et al. 1994
Communicating Process	HOARE 1979, EBERGEN 1991, MARTIN 1990
Tangram	van BERKEL 1993
Micropipelines	SUTHERLAND 1988, 1993
Change Diagrams	KISHINEVSKY 93, SCHWIEGELSHOHN 1998
Biased concurrent STG	WOLLOWSKI 1998
Slots	SIMSON 1991, XIA 1998
Hardware Petri Nets	ROKITA 1998
VHDL asynchrone	PROTEROE 1998, MUTZ 1998
Communicating Programs	BRUNDVAND 1991, 1993
Dynamic Clocking	DEAN 1992
StarCarts	KOL 1996
SpecC	GAJSKI 1998

Une première classification par primitive de calcul (P. de Calcul) et par primitive de spécification (P. de spécification) est montre par la table ci-après:

NOM DU MODÈLE	P. DE CALCUL	P. DE SPEC
FSM asynchrone	à états	FSM
Machines auto-synchronisées	à états	FSM1
Machine à données	à états	FSM
FSM avec horloge locale	à états	FSM
Réseaux de Petri et graphe des données	à événements	PN avec données
Machine en mode rafale étendu	à états	FSM
Asynchronous Logic Arrays	à événements	PN
Graphes de signal	à événements	Graphe marqué
I nets	à événements	PN
Signal Transition Graphs STG	à événements	PN
State Graphs	à états	-
Communicating Processes	à processus	CSP
Tangram	à processus	CSP
Micropipelines	à données	pipeline
Change Diagrams	à événements	Graphe de changements
Biased concurrent STG	à événements	STG
Slots	à rôles	CPN
Hardware Petri Nets	à événements	CPN
VHDL asynchrone	à événements	PN, RTL
Communicating Programs	à processus	Occam
Dynamic Clocking	gestion de l'horloge	-
StarCarts	à états	StateCharts
SpecC	à états	Extension à C

La spécification de circuits et systèmes asynchrones suit une approche modulaire. Le système est spécifié comme un réseau de blocs. Nous nous intéressons

aux méthodes de synthèse architecturale et logique pour évaluer leurs liens avec la conception système et architecturale. Les méthodes automatiques qui facilitent ces tâches sont appelées synthétiseurs ou traducteurs de programmes à circuits. La table ci-après montre l'état actuel de ces outils.

NOM DU PROGRAMME	TYPE DE CIRCUIT	AUTEUR ET DATE
CSP	Circuits autosynchronisés	BURNS 88
OCCAM(basés CSP)	Circuits DI	BRUNVAND 91
TANGRAM(Basés CSP)	Circuits 'Handshake'	van VERKEL 93
OCCAM asynchrone	circuits CMOS asynchrones	NEDELCHEV 95
VHDL comportemental	micropipelines	TAN 1998

Cette analyse nous montre que la synthèse des circuits asynchrones atteint la maturité [50], [62], [51], [52], [53], [55], [56], [57], [58], [59], [60], [61], [63]. La plupart des outils CAO synthétisent ces circuits à partir de deux spécifications : les machines à états en mode rafale et les STG. Les circuits synthétisés à partir de ces formalismes sont exempts d'aléas mais ils sont différents par rapport à leurs hypothèses sur le modèle de retard utilisé (mode de fonctionnement fondamental contre mode de fonctionnement indépendant de la vitesse ou "Speed Independent").

Nous allons étendre la modélisation de systèmes et des circuits asynchrones au-delà de la synthèse architecturale et logique vers la modélisation du niveau système et nous allons retenir le modèle des réseaux de Petri interprétés [9] comme outil de modélisation mathématique pour combler la distance entre ces deux niveaux d'abstraction. Ce choix a été orienté par leur usage généralisé, parce qu'ils modélisent les systèmes asynchrones d'une façon naturelle, par leur puissance d'expression et par la facilité de leur utilisation.

1.7 Conclusions

Pour la phase d'analyse et de spécification des systèmes, une approche fonctionnelle facilite la compréhension du problème et sa formalisation. Cette phase

initiale du cycle de vie s'appuie sur le cahier des charges qui exprime les besoins sous forme de fonctions à réaliser, de services à rendre et de contraintes à respecter. Une spécification fonctionnelle est naturelle. Elle exploite directement les données fournies par le cahier des charges pour l'élaboration d'un modèle logique qui dégage les fonctionnalités du système et ses contraintes. Ces dernières sont directement exploitables par le client pour une validation préliminaire.

Pour les phases suivantes du cycle de vie des systèmes, les approches fonctionnelles montrent des limites en ce qui concerne la réutilisation, la maintenance, la mise à jour ainsi que les adaptations et les évolutions.

Les approches orientées objets résolvent ces inconvénients en introduisant des concepts nouveaux d'abstraction de données, d'héritage et de polymorphisme qui améliorent les qualités des réalisations obtenues.

Néanmoins, aucun des modèles, méthodes et méthodologies présentés ne fournit un lien hiérarchique vers la conception en aval (modèles physiques et modèles de défaillance) en vue d'une réalisation sur des circuits asynchrones [64] et aucun des modèles présentés pour la synthèse logique et comportementale de circuits asynchrones ne fournit de lien vers la conception amont (acquisition des exigences, définition des spécifications, conception architecturale).

Dans le cadre des applications envisagées, une approche à plusieurs modèles cohabitant dans le même environnement de conception est nécessaire, et ceci pour la modélisation des différents domaines (flots de matière, d'énergie et d'information) dont les Micro Systèmes sont le siège et la modélisation des domaines transverses tels que l'évaluation de la faisabilité économique et technologique [2].

Nous avons choisi la méthode SA-RT, le modèle des réseaux de Petri Interprétés et les langages SDL et VHDL-AMS comme outils de base pour la méthode, le modèle et le langage que nous allons proposer dans le chapitre suivant. Ce choix a été orienté par les critères ci après:

- une puissance suffisante pour représenter systèmes complexes avec un espace d'état assez large et multidisciplinaire;
- une simplicité suffisante pour pouvoir synthétiser des circuits électroniques et électromécaniques;
- la formalisation des concepts tels que la concurrence, la causalité, le choix entre événements et le parallélisme au niveau système;
- la modélisation mixte des aspects statiques (structuraux), des aspects dy-

namiques (comportementaux), des aspects événementiels et des aspects de traitement des données;

- l'intégration de différents langages de modélisation;
- la représentation du système à différents niveaux d'abstraction et ;
- l'animation et la simulation des représentations;
- la disponibilité des outils et de normes pour les construire, les analyser et les simuler.

Chapitre 2

LA CONCEPTION AMONT (SPECIFICATION ET VALIDATION ARCHITECTURALE)

2.1 Introduction

Nous allons proposer dans ce chapitre une méthode descendante de conception amont, un modèle de représentation assurant la formalisation dès le début de la conception, et un langage de spécification architecturale de haut niveau, se situant à l'intérieur du paradigme méthodologique exposé dans le chapitre précèdent.

Les besoins les plus importants exigeant une telle approche sont :

- L'information de la conception doit circuler entre les différents outils CAO et les concepteurs d'une manière facile, souple et efficace, et ceci doit faciliter la synthèse et l'analyse concurrents ;

- Visualisation et sélection des composants candidats à être réutilisables

(Blocs "Intellectual Property - IP ") [67], [68], [69], [70] à partir de différentes sources telles qu'Internet;

- le développement d'un dictionnaire extensible de termes standards pour véhiculer l'information croissante de conception;
- Le développement des mécanismes de distribution sûre des résultats de la conception sur Internet;
- un environnement informationnel y compris un système de gestion des bases de données utilisant une interface commune et utilisable pour tous les outils de l'entreprise;
- une amélioration des systèmes de gestion des bases de données et des processus de conception;
- l'intégration de la conception de haut niveau (système), de la synthèse et de la modélisation physique;
- permettre les nécessités des niveaux inférieurs en utilisant l'estimation de haut niveau des quantités technologiques et physiques en rapport avec la performance des portes et des transistors et leurs interconnexions;
- assurer que les architectures décrites sont correctes par leur construction, en vue de minimiser les coûts de vérification.

Dans ce chapitre, nous allons présenter une méthode structurée descendante globale applicable aux domaines des Micro Systèmes électromécaniques, mais également extensible à d'autres types de Micro Systèmes. D'autre part, un logiciel de spécification de haut niveau a été développé, HiLeS (High Level Specification Software), basé sur les Réseaux de Petri Interprétés et écrit en langage Java. Ce logiciel permet la synthèse de fichiers en langage VHDL-AMS à partir de la simulation des échanges entre les blocs qui composent le Micro Système, l'environnement et l'utilisateur.

Comme nous l'avons indiqué précédemment, les deux difficultés majeures concernent le format des fichiers transmis d'une phase à l'autre du processus ainsi que la nature des données (analogiques, numériques, mécanique,...) qui transitent d'un sous-ensemble à un autre à l'intérieur du système à concevoir. Les solutions adoptées pour résoudre ces problèmes ont été d'utiliser la norme

VHDL-AMS avec l'adaptation de passerelles pour assurer la continuité nécessaire. la suite, nous nous sommes inspirés de la théorie des systèmes vivants pour assurer la compatibilité des échanges et une classification hiérarchique lors de la constitution des sous-ensembles du dispositif à concevoir.

2.2 La Méthode de Conception Descendante HiLeS

Dans une approche descendante, la complexité est maîtrisée en utilisant la décomposition et la division. Les micro-systèmes sont décomposés en éléments plus petits, cohérents et autonomes et les fonctions, les processus et les structures sont divisés en sous-systèmes, sous sous-systèmes, etc.. Pour chaque élément et sous-système, un rapport doit être produit. Les résumés de chaque rapport constituent ensemble le dossier technique du projet.

La méthode de conception suggérée ici (voir figure 2.1) suit une approche basée décision [45] et sur la standardisation ANSI/EIA-632-1998 [47]. Nous avons appellée cette méthode: La méthode HiLeS (High Level Specification). Les tâches dans le processus de conception sont groupées en modules. La sortie de chaque module est une décision; ainsi la conception de l'architecture pour un problème donné devient une série de décisions. Dans ce contexte, le rôle principal d'un ingénieur est de prendre des décisions associées à la conception, à la fabrication et à la mise en opération des Micro Systèmes. Dans cette approche, le processus entier, de la conception à la fabrication, est traité comme unité pour fournir une solution. La méthode adoptée permet de prendre des décisions telles que:

- la sélection, basée sur des critères multiples, d'une solution faisable,
- la modification d'une solution pour l'améliorer et,
- la décision basée sur les connaissances de veille technologique.

Le coeur de la méthode est la génération des descriptions architecturales, que ce soit pour une esquisse initiale (acquisition des exigences), ou pour une description plus détaillée en vue de la validation structurelle et fonctionnelle. Le détail de la méthode est montré par la figure 2.2.

Nous avons défini un modèle et un langage de description de haut niveau (Modèle et Langage HiLeS). Leur utilisation est à la base de la communication entre concepteurs et permet l'automatisation de la méthode.

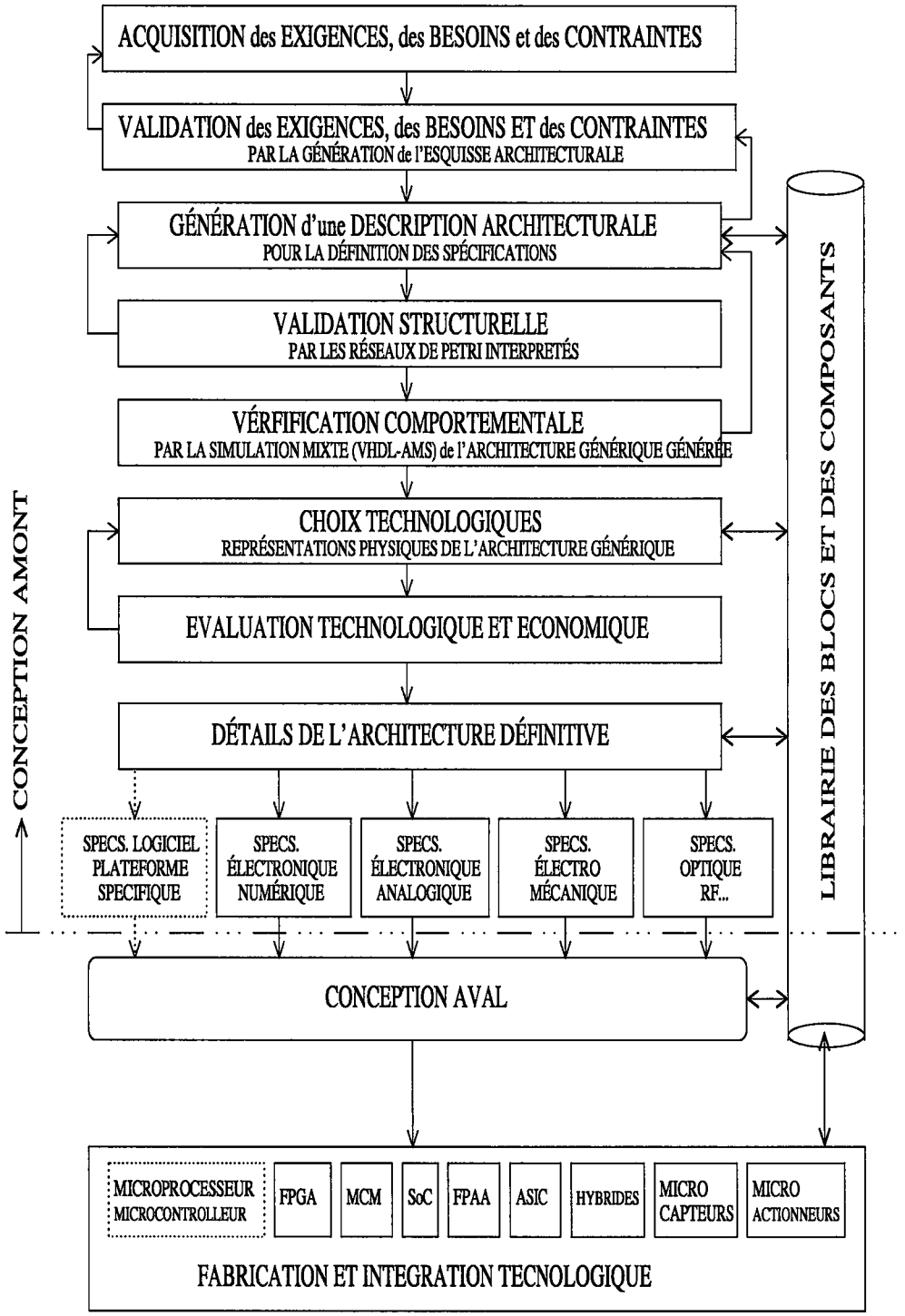

FIG. 2.1 – *Méthode HiLeS pour la Conception Amont de Micro Systèmes*

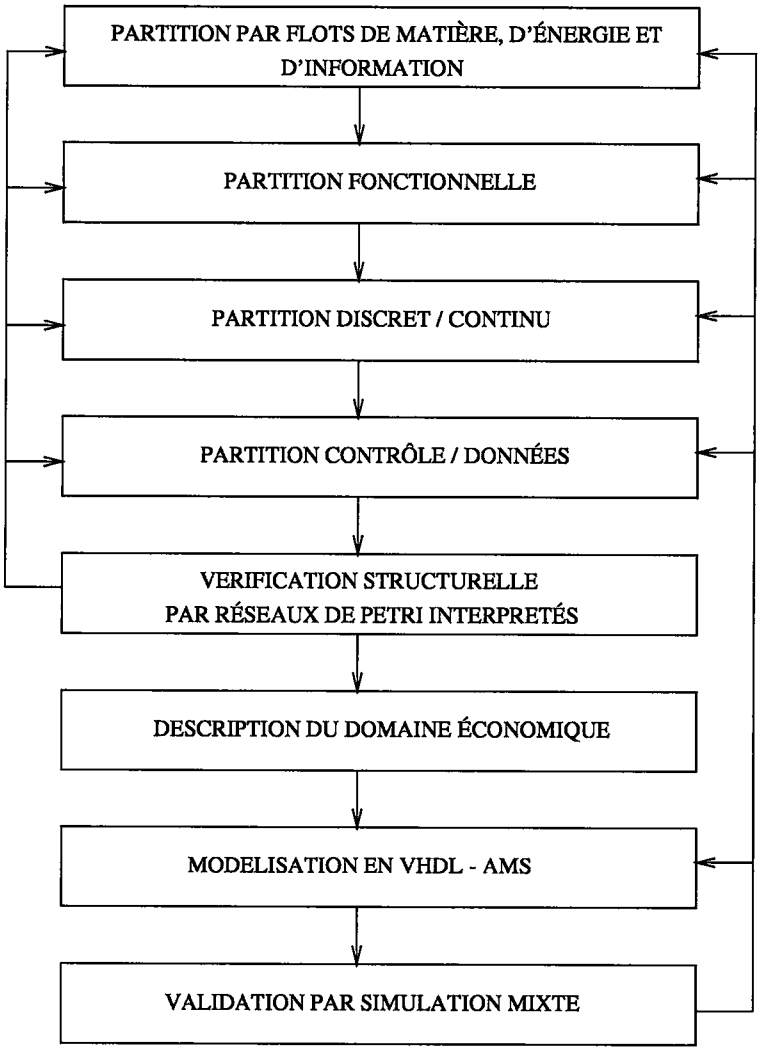

FIG. 2.2 – *Méthode pour la Génération de Descriptions Architecturales*

Nous allons tout d'abord définir le modèle HiLeS et par la suite nous entrerons dans le détail de la méthode.

2.3 Le Modèle et le Langage de Spécification et de Description HiLeS

2.3.1 Objectifs.

Le langage de spécification et de description (HiLeS) permet de spécifier et de décrire, d'une manière non ambigüe, le comportement et l'architecture des Micro Systèmes et des réseaux de télécommunication des Micro Systèmes. HiLeS est une extension de la norme CCITT Z.100 (Specification Description Language SDL) [3], [4].

La définition de HiLeS a comme objectif de fournir un modèle (et un langage) formel, facile à utiliser, à interpréter et à apprendre ; il est assez puissant et général sur une famille d'applications qui suivront ; il fournit des descriptions non ambigües en vue d'une réalisation par des circuits asynchrones; il est indépendant de la technologie de fabrication et il facilite la synthèse descendante des applications visées.

Le domaine d'application de HiLeS est la description de systèmes complexes distribués fonctionnant en temps réel. En particulier, les applications listées ci après sont spécialement visées : systèmes d'enregistrement de contraintes de systèmes complexes, systèmes de supervision et diagnostic, systèmes de maintenance et de surveillance du vieillissement ; systèmes de contrôle distribué. HiLeS permet la description d'un objet pourvu que cet objet puisse être décrit par un modèle hybride (discret/continu). HiLeS peut être utilisé pour faire des spécifications informelles de haut niveau, semi-formelles, formelles incomplètes et détaillées. Il peut être utilisé pour définir l'expression des besoins, la conception architecturale, la spécification détaillée, la synthèse architecturale, la spécification de la fiabilité, la validation et la vérification.

2.3.2 Définitions de base

HiLeS propose deux formes syntaxiques différentes : Une représentation graphique et une représentation textuelle. Les deux formes sont équivalentes à une sémantique unique. La définition formelle de HiLeS permet la transformation

d'une spécification, représentée avec une syntaxe graphique ou une symbolique, à grammaire unique. Les définitions des concepts généraux et conventions utilisés par HiLeS sont présentées maintenant.

Composants

Les composants n'existent que par leur définition. La définition comprend toutes les propriétés qui lui sont associées.

Environnement

Les systèmes spécifiés par HiLeS sont réactifs aux stimuli reçus de l'environnement. Le comportement de l'environnement est non déterministe et il est spécifié par des processus différents des processus du système.

2.3.3 Structure du Système

Un système HiLeS est composé d'un ensemble de blocs, les blocs étant interconnectés entre eux et l'environnement par des canaux. A l'intérieur de chaque bloc il peut exister d'autres blocs et des processus. Les blocs communiquent entre eux et avec les processus par l'intermédiaire de signaux et ils agissent concurremment.

Système

La définition du système doit contenir les définitions de tous les canaux, signaux, types de données qui existent entre l'environnement, les blocs et les réseaux de contrôle du système. Une définition du système doit contenir au moins une définition de bloc. Les canaux transportent des signaux auxquels sont associées des données ou des jetons. Les données référencées dans la spécification et les signaux sont définis dans la partie déclaration. Un système est un arbre n-aire où les sommets sont des blocs et les feuilles sont des processus. Les canaux relient les blocs du même niveau de la hiérarchie. La partie déclaration ne contient pas de définition de variables étant donné que chaque processus possède sa propre mémoire locale.

Bloc

Un bloc contient un ou plusieurs blocs et / ou un ou plusieurs processus. Il a pour but de regrouper des sous structures qui se comportent comme une unité

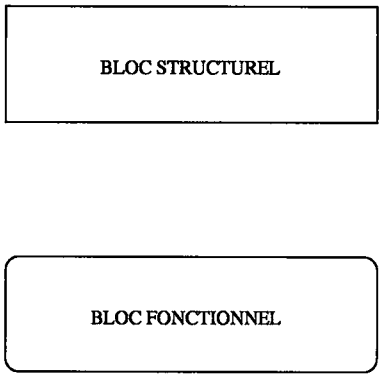

FIG. 2.3 – *Blocs Structurels et Fonctionnels*

fonctionnelle et qui agissent directement ou à travers une autre sous structure. Son comportement est spécifié par un processus. La définition de bloc fournit l'interface de communication statique entre sous structures et établit le cadre pour leur définition.

Il existe deux types de blocs :

- d'une part les blocs structurels, représentés par des rectangles dans la grammaire graphique ;
- d'autre part les blocs fonctionnels, représentés par des rectangles avec les angles arrondis, dans la grammaire graphique.

Les blocs structurels servent à spécifier la structure d'une manière hiérarchique. Les blocs fonctionnels servent à définir un comportement par une équation logique, algébrique ou différentielle (Figure 2.3). Les blocs fonctionnels ont des délais associés équivalents à la durée d'exécution de la fonction si elle est statique ou au temps actuel si elle est dynamique. Par exemple, un amplificateur opérationnel, un capteur ou une fonction combinatoire peuvent être représentés par ces blocs.

Un Processus

Un processus est un réseau de Petri ordinaire, interprété et sauf auquel est ajouté un graphe de données. Le réseau de Petri est appelé réseau de contrôle. Le graphe de données peut être exprimé comme une structure formée par des blocs fonctionnels reliés par des canaux. Le réseau de Petri, par le flot de jetons,

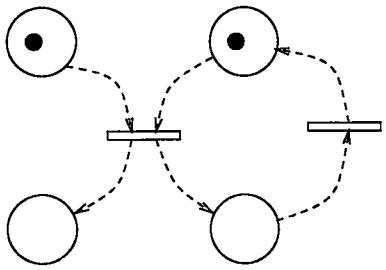

FIG. 2.4 – *Réseau de Contrôle (Réseau de Petri Interprété Sauf)*

contrôle et séquence l'exécution (événements et activités) des fonctionnels. La durée du tir des transitions peut être différente de zéro (Figure 2.4).

2.3.4 Communications

Canaux

Un canal représente une route pour le transport des signaux. Un canal peut être considéré comme une ou deux trajectoires unidirectionnelles et indépendantes entre deux blocs ou entre un bloc et un réseau de contrôle. Les signaux transportés par le canal sont délivrés au destinataire au point terminal du canal. Les signaux sont délivrés au destinataire dans le même ordre où ils sont présentés au point d'origine. Si deux signaux arrivent simultanément ils seront ordonnés arbitrairement. Les canaux peuvent introduire des retards aux signaux. Il existe donc une queue du type FIFO associée à chaque direction de transmission. Trois types de canaux sont définis (Figure 2.5):

- Les canaux à jetons sont représentés par des flèches vides unidirectionnelles et à traits pointillés (grammaire graphique). Ces canaux transportent les jetons et correspondent aux arcs des réseaux de Petri. Les jetons ne sont pas colorés et ils informent les blocs de l'occurrence des événements.

- Les canaux à événements sont représentés par des flèches vides à traits continus (grammaire graphique). Ces canaux représentent les flots des informations par événements. Ces canaux transportent des signaux qui sont discrets dans le temps.

- Les canaux continus sont représentés par des flèches pleines et à traits continus (grammaire graphique). Ces canaux transportent des informa-

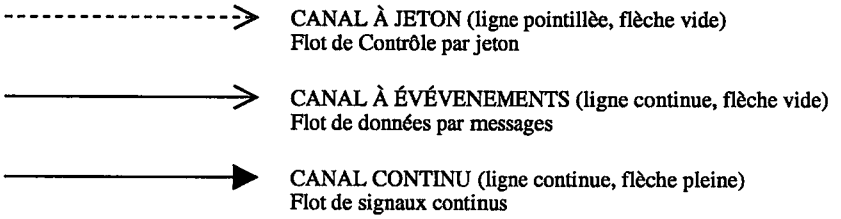

FIG. 2.5 – *Définition des canaux*

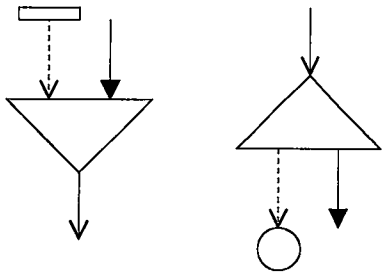

FIG. 2.6 – *Opérations de blocage et échantillonnage*

tions continues (analogiques ou numériques) , donc les signaux transportés sont continus dans le temps.

Les opérations d'échantillonnage et de blocage sont définies pour permettre la conversion de canaux. Ainsi, l'échantillonnage d'un canal continu produit un canal à événements. Des triangles avec un arc et un canal continu à l'arrivé et un canal à événements en sortant représentent l'opération d'échantillonnage (grammaire graphique). D'autre part, le blocage d'un canal à événements produit un canal continu et un arc en sortie. Des triangles avec un canal à événements à l'arrivée et avec un canal continu et un arc à la sortie représentent l'opération de blocage (grammaire graphique). L'opération de blocage équivaut à une structure de mémoire qui maintient sur le canal continu une information arrivant sur le canal à événements. Les blocs structuraux peuvent recevoir et émettre tout type de canal ; les blocs fonctionnels peuvent seulement recevoir et émettre des canaux continus (Figure 2.6).

Les canaux continus et à événements peuvent transporter un (ligne mince) ou plusieurs (ligne épaisse) signaux et de différents types. Les signaux représentent des données. Les types de données définis sont : Binaire, Entier, Virgule flottante, Symbole, Vecteur et Structure.

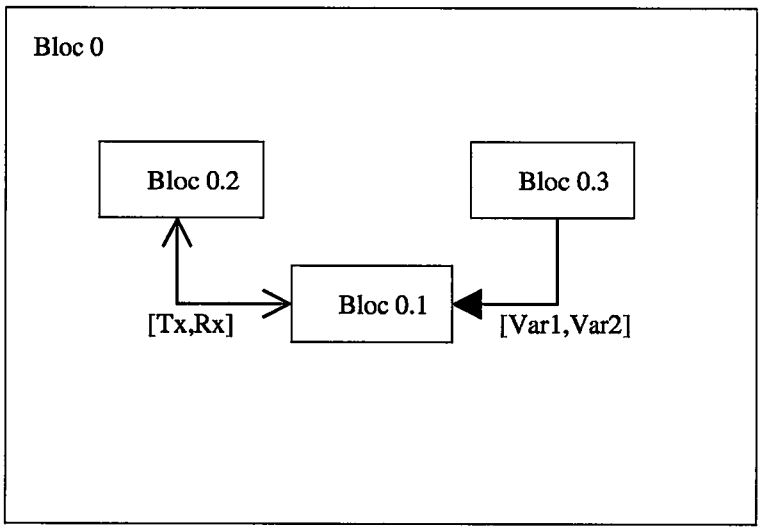

FIG. 2.7 – *Exemple de définition de système*

La Figure 2.7 montre un exemple de définition d'un système et de son environnement. Les signaux discrets Tx et Rx s'échangent entre le système et son environnement 1. Les signaux continus Var1 et Var2 voyagent entre le système et son environnement 2. La Figure 2.8 montre la définition d'un bloc. Le bloc est composé d'un réseau de contrôle, une fonction, une sous structure et des canaux de communication.

Communication données - contrôle

Les flots de données et de contrôle sont indépendants et concurrents. Les réseaux de Petri représentent les flots de contrôle. Les structures et les fonctions communicantes représentent les flots de données. Le contrôle et les données peuvent communiquer à travers des opérations de blocage et d'échantillonnage. Néanmoins, un processus peut être initialisé par un ordre de contrôle ou une séquence de contrôle peut dépendre de la condition d'une donnée. Des canaux continus entrant et sortant du réseau de contrôle sont proposés pour représenter l'interaction entre les données et le contrôle. Un signal binaire sortant d'une place a la valeur 'vrai', si la place contient un jeton et elle a la valeur 'faux' sinon. Un signal binaire de valeur 'faux' entrant sur une transition interdit le tir de la transition. La valeur 'vrai' du signal autorise le tir correspondant. Un signal binaire sortant d'une transition prend la valeur vrai pendant le tir de la

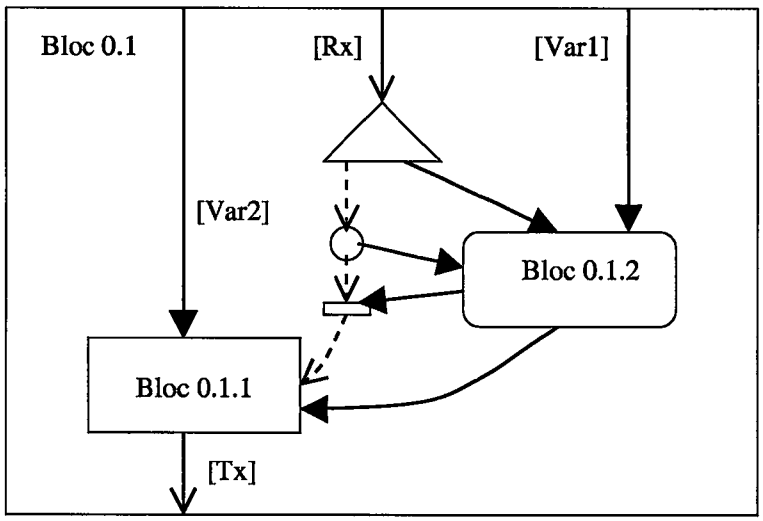

FIG. 2.8 – *Exemple de définition de bloc*

transition.

La Figure 2.9 montre un réseau de Petri qui communique la valeur binaire 'A' à une fonction combinatoire. 'A' est vrai pendant que le jeton sejourne dans la place respective et faux autrement (dans le sens de test à l'état d'une place). Outre 'A', la fonction a trois entrés (B, C et D) et une sortie 'E'. La sortie binaire 'E' est 'vrai' si 'A' est 'vrai' et que $'C < B'$ est 'vrai' ou elle est 'faux' si 'A' est 'faux' et que $'D > B'$ est 'vrai'. Cette sortie peut interdire le tir de la transition du réseau de Petri droite de la figure si sa valeur est 'faux' ou sinon l'habiliter.

Un signal transportant une valeur en virgule flottante peut entrer dans une transition. Dans ce cas, la valeur est interprétée comme un retard pendant lequel la transition ne peut pas être tirée même si elle est validée. Ce retard est utilisé pour synchroniser le tir de la transition avec la terminaison d'une opération. Dans le cas où le temps d'opération serait variable, le retard correspond au retard du pire des cas, si un circuit de détection de la terminaison de l'opération n'est pas spécifié.

La Figure 2.10 montre la spécification d'un compteur de signaux '1' dans un message binaire donné. Le bloc fonctionnel a trois fonctions. La fonction 'O' décale d'un bit vers la droite le message 'I' ; la fonction 'Inc' est le bit de moindre poids ; 'Fin' est 'vrai' si 'I' est zéro ; 'Temp' est le retard maximal du

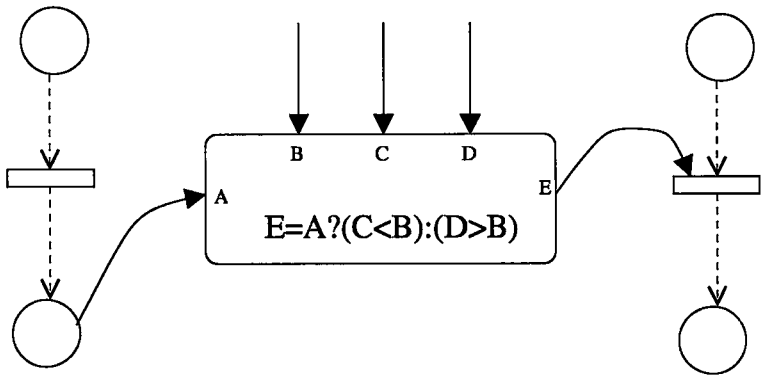

FIG. 2.9 – *Communication entre données et contrôle*

bloc. 'Cnt' avance le compteur à chaque arrivée d'un jeton sur '+1' et s'efface à chaque arrivée d'un jeton sur '=0'. Le jeton est rendu lorsque l'opération est finie.

2.3.5 Relations physiques

Les relations physiques entre les blocs sont à déterminer dès le début de la conception d'un micro système. Des questions telles que :

- où va être situé le micro système?
- quels sont les conditions environnementales que le micro système doit subir?
- va-t-il être mobile ou fixe?
- quels doivent être le support ou le boîtier du micro système?

Une relation physique est représentée par une ligne continue avec un losange à chaque extrémité. Ce symbole relie les deux blocs, ayant une relation physique.

2.4 Caractéristiques des Architectures Décrites

Une architecture consiste en la spécification des composants d'un Micro Système et de leurs communications. Le Micro Système doit être conforme à l'architecture. Dans ce sens, l'architecture doit garantir certaines propriétés du comportement du Micro Système et elle doit être utile pendant le processus

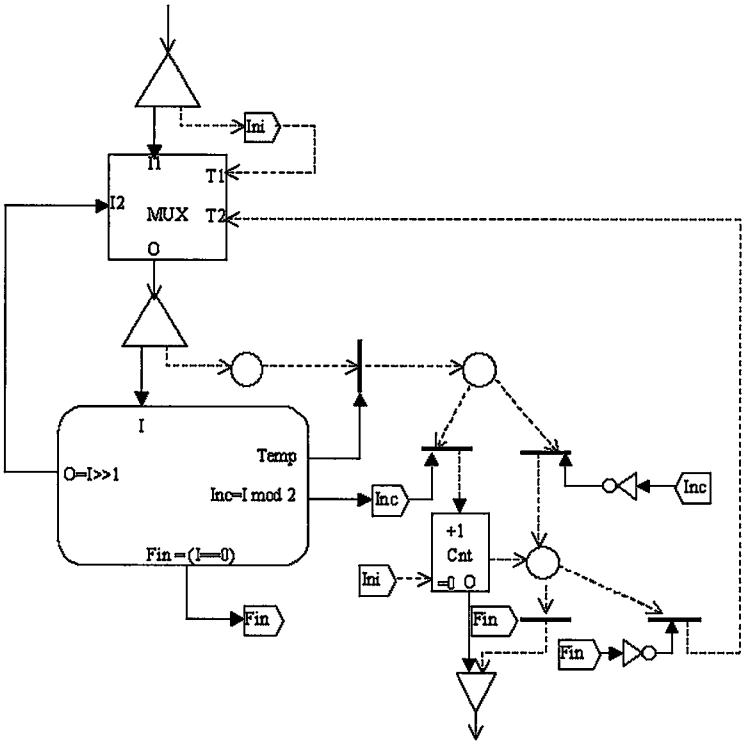

FIG. 2.10 – *Définition d'un compteur des '1' dans un message binaire donné*

de construction. Une architecture décrite en HiLeS est une architecture orientée connexions pour laquelle le critère de l'intégrité de la communication est essentiel pour garantir des communications correctes entre composants. Par conséquent, une architecture orientée connexions doit avoir les propriétés suivantes :

- interface idéale,
- connectivité des interfaces et
- connexions correctes.

Déterminer si une architecture possède ces propriétés est un problème non décidable en général.

Nous allons montrer que la définition de HiLeS permet de gérer des architectures complexes (modularité et hiérarchie), de faire du raffinement de la description et de garantir les propriétés du Micro Système par simple inspection de l'architecture, grâce au fait qu'il respecte les propriétés ci dessus.

2.4.1 Interfaces idéales

Une interface doit assurer la capacité de substitution des blocs qui lui sont conformes. Par conséquent et selon le principe d'interface idéale[1] [44], l'interface d'un bloc spécifie cinq caractéristiques :

- la liste de signaux (données ou contrôle) fournis par le bloc,
- la liste de signaux (données ou contrôle) requis par le bloc,
- la spécification du comportement des signaux fournis par le bloc,
- la spécification du comportement des signaux requis par le bloc,
- la spécification de l'interaction mutuelle entre signaux fournis et requis.

Un service est tel qu'une interface est structurée comme étant des groupes de signaux (requis et fournis) chacun avec ses propres contraintes formelles. La communication entre les blocs d'un système est spécifiée par un ensemble de connexions entre les services des blocs. Toute communication externe d'un bloc se fait par ses services. Un protocole spécifie l'ordre (séquence) du flot de données et de contrôle entre les services.

1. " If the interface of module x is the same as the interface of module y, then for any program P which contains the module x and can be shown to behave in some required manner on the basis of the information in the interface alone, the module y could be substituted for x and the resulting program would satisfy the same behavioural requirements "

2.4.2 Connectivité des processus et connexions correctes

La connectivité de deux services est une propriété facile à tester et suffisante pour assurer une connexion correcte. Une connexion correcte entre deux services est telle que le transfert de données et d'information est consistant avec toute contrainte sur les signaux: contraintes fonctionnelles (relations fonctionnels), les contraintes de domaine (séquences et domaines appropriés des entrées) et les relations du type *entrée précède sortie, sortie précède entrée et sortie précède sortie.* Une connexion correcte d'un signal est correcte s'il s'agit d'un signal requis et un signal fourni du même type et avec les mêmes contraintes formelles (type, par exemple). La connexion correcte doit satisfaire, aussi, le protocole défini pour ordonner le flot de données et de contrôle entre services.

2.4.3 Intégrité de la communication

L'intégrité des communications implique que toute communication entre composants (blocs) du système soit spécifiée par les connexions de l'architecture. Seules les communications entre blocs sont contraintes et non pas les communications internes au bloc. L'approche adoptée correspond à la stricte visibilité de l'interface. Cela signifie qu'un composant du système peut interagir avec d'autres composants seulement en utilisant sa propre interface. Si un processus d'un bloc est utilisé par plusieurs blocs alors le processus est dit partagé et il est nécessaire que ce partage soit sûr. Un bloc est partagé de façon sûre si tous les blocs concernés lui sont connectés. Cette contrainte oblige le concepteur à définir complètement les communications du système de manière à avoir des connexions sûres de fonctionnement.

2.4.4 Services de l'interface

HiLeS définit une notation spécifique pour les connexions de service en utilisant des noms de service. Une connexion de service permet d'encapsuler dans un canal une liste de services. Un service est représenté par un rectangle à l'arrivé des canaux.

Une connexion de service est un canal entre deux services compatibles. Deux services compatibles sont deux services, tels que tout signal fourni par un service correspond à un signal requis dans l'autre service et ils peuvent être connectés correctement.

La sémantique d'un processus d'interface et d'une connexion est :

```
<SERVICE nom_du_service>
  <SIGNALS_P nom_de_signal:type_de_signal><\SIGNALS_P>
  <SIGNALS_R nom_de_signal: type_de_signal><\SIGNALS_R>
          <CONSTRAINT nom_du_protocole><\CONSTRAINT>
<\SERVICE>

<CHANNEL nom_du_canal
  FROM nom_du_bloc|nom_du_service TO nom_du_bloc |
<SIGNALS nom_du_signal:type_du_signal><\SIGNALS>
<\CHANNEL>
```

Un exemple de connexion correcte est montré ci après:

```
<BLOCK petri_memo IS PN MODEL = petri_memo.vhd
                      KEYCHARACTERISTICS =a,b,c,d,e,f>
      <SERVICE a> <SIGNALS_R
                      leer_mem:TOKEN>
                <\SIGNALS_R>
      <\SERVICE>
      <SERVICE b> <SIGNALS_P
                      read:BIT >
                <\SIGNALS_P>
                <SIGNALS_R
                      ack_mem:BIT>
                <\SIGNALS_R>
      <\SERVICE>
      ...
      <BLOCK p1 IS PLACE><\BLOCK>
      <BLOCK p2 IS PLACE><\BLOCK>
      <BLOCK t1 IS TRANSITION><\BLOCK>
      ...
      <CHANNEL c1 FROM a TO  p1>
                <SIGNALS
                      leer_mem:TOKEN>
```

```
                <\SIGNALS>
        <\CHANNEL>
        <CHANNEL c2 FROM p1 TO  t1> <\CHANNEL>
        <CHANNEL c3 FROM t1 TO  p2> <\CHANNEL>
        <CHANNEL c7 FROM p1 TO  b>
                <SIGNALS_P
                     read:BIT>
                <\SIGNALS_P>
        <\CHANNEL>
        <CHANNEL c8 FROM b TO  t1>
                <SIGNALS_P
                     ack_mem:BIT>
                <\SIGNALS_P>
        <\CHANNEL>
        ...
<\BLOCK >
```

La figure 2.11 montre la version graphique des connexions décrites ci avant.

2.4.5 Librairie des blocs

Chaque bloc de la bibliothèque est caractérisé par son identification, sa classe, ses différents modèles, ses caractéristiques clés et ses services (interface physique).

Interface physique

L'interface physique d'un bloc détermine la nature physique de ses ports d'interconnexion, qui peut être:

- mécanique
- optique
- chimique
- électromagnétique
- thermique
- électrique

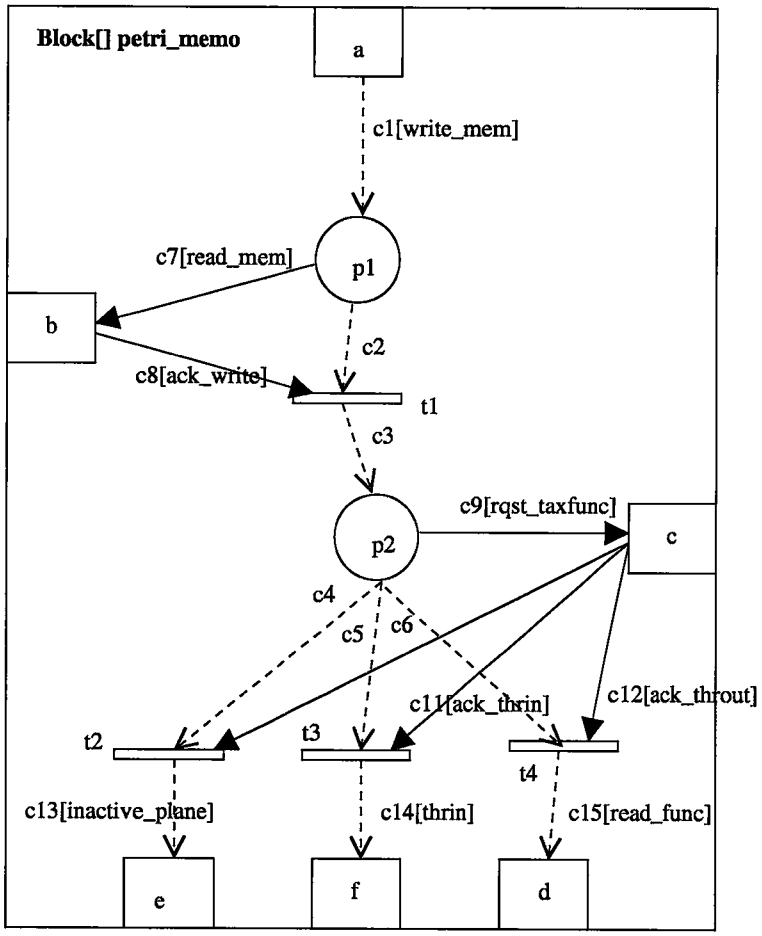

FIG. 2.11 – *Exemple d'un Bloc de Contrôle*

- numérique
- fluidique

Des blocs d'interfaces physiques différentes ne peuvent pas être reliés directement sans un convertisseur approprié. De manière générale, les aspects continus du comportement des blocs sont modélisés par un système ordinaire d'équations différentielles ou algébriques (EDA) dont le temps est la variable indépendante. Les variables inconnues du système de EDAs sont des fonctions analytiques du temps (fonctions continues à morceaux avec un nombre fini de discontinuités) et elles correspondent au concept des *Quantités* de VHDL-AMS. Ainsi, les définitions des quantités doivent apparaître explicitement dans la définition des services des blocs (voir 2.4.4).

Classes de blocs

Les classes de blocs correspondent aux différentes manières de traiter l'énergie, la matière ou l'information [46]. Ce choix fournit une indépendance vis à vis de la nature physique du composant. Ces classes donnent aussi une méthode pour élaborer la partition fonctionnelle du Micro Système et facilitent l'assignation des exigences à des sous structures typiques. Les classes de blocs dépendent du niveau d'abstraction. Les niveaux d'abstraction définis sont :

- cellule
- organe
- organisme
- groupe
- organisation

Ainsi, un bloc du niveau cellule est un élément de base d'un bloc du niveau organe ; un bloc du niveau organe est un composant d'un bloc du niveau organisme; un bloc du niveau organisme est un Micro Système individuel; un bloc du niveau groupe est un ensemble de Micro Systèmes communicants; et un bloc du niveau organisation est un groupe de Micro Systèmes organisés selon une architecture adaptée au problème (hiérarchie, réseau, etc.). Pour faciliter la conception d'un organisme et des systèmes de niveau d'abstraction supérieur (groupe et organisation), quince classes d'organes ou de blocs du niveau organe

ont été définies. Nous avons choisi cette structure en analogie avec la théorie des systèmes vivants étant donné qu'un Micro Système est fait pour fonctionner toujours immergé dans un environnement naturel auquel il doit s'intégrer.

Les classes retenues sont :

- capter: acquérir des quantités physiques de l'environnement (capteurs, transducteurs, détecteurs, suiveurs.
- ingérer: ingestion des flots de matière (tuyaux, bouches, canaux)
- expulser: expulsion des flots de matière (tuyaux, puits, canaux)
- conditionner: transformation des signaux et des quantités physiques en d'autres formes utiles (amplificateurs, filtres, transformateurs).
- traiter: traitement de l'information (codeurs, décodeurs, multiplexeurs, comparateurs, fonctions d'association, diagnostique, surveillance, coordination)
- mémoriser: mémorisation d'information pendant des périodes de temps spécifiques (FIFO, RAM, flash EEPROM)
- temporiser / dater: mise à jour des informations relatives au temps de l'environnement (temps universel) ou des blocs (UTC, temporisateurs).
- décider: prise de décision sur des actions à suivre (micro contrôleurs, micro processeurs, programmateurs)
- commander: séquencement des activités de commande (séquenceurs, contrôleurs)
- actionner: modification des relations physiques entre blocs (moteurs, chaînes de test, actionneurs)
- alimenter: stockage d'énergie pendant des périodes de temps spécifiques (piles, accumulateurs).
- interconnecter: transport des signaux et des quantités physiques entre blocs (canal, bus, réseau).
- recevoir / émettre: recevoir et transmettre de l'information (émetteurs, récepteurs, modems, ponts).

- supporter : maintien des relations spatiales spécifiques entre blocs ou entre le bloc et son environnement (circuits monolithiques (SoC), circuits multipuce(MCM)).
- empaqueter : séparation du bloc de son environnement, regrouper ses constituants ensemble, les protéger et autoriser ou non l'entrée ou la sortie de différents types de matière ou d'énergie (packaging).

Modèles des blocs

Un bloc de la librairie peut être décrit par un ou plusieurs modèles. Le but cette caractérisation est de composer des descriptions multiples d'un composant unique, de sélectionner une parmi plusieurs représentations du composant et de combiner des multiples représentations pour décrire un composant.

Pour répondre à ces objectifs, nous avons défini des opérateurs binaires pour que la liste de descriptions multiples d'un composant devienne alors une définition de leurs interactions.

La composition des modèles L'opérateur *ET* spécifie la simultanéité des modèles interagissant. Les modèles mis en rapport par cet opérateur doivent être simultanément valides et mutuellement consistants.

La disjonction des modèles L'opérateur *OU* spécifie l'exclusion mutuelle entre les modèles interagissant. L'interaction entre modèles disjonctifs devrait forcer l'exclusion mutuelle, mais elle n'est pas garantie. Cet opérateur a pour but de définir des familles de composants, des modélisations à différents niveaux d'abstraction et encore des modes opérationnels différents.

Description des Composants du type organe

Toutes les informations relatives aux composants et sous-systèmes disponibles dans le commerce ou développés en interne sont représentées à l'aide d'une déclaration de bloc de la manière suivante :

```
<BLOC   nom du Composant  IS ORGANE classe de l'organe
 MODEL = modèle.vhd
   FABRICANT =
   DOMAINES  D'APPLICATION  =
   MATERIAUX ET TECHNOLOGIES CLES =
```

```
ETAT INDUSTRIEL =
INTERFACES.OPTIQUE =  Fibre Optique | Libre
INTERFACES.MECANIQUE & THERMIQUE = type d'interface
INTERFACES.ELECTRIQUE = type d'interface
INTERFACES.ANALOGIQUE = type d'interface
INTERFACES.NUMERIQUE = type d'interface
INTERFACES.ELECTROMAGNETIQUE = type d'interface
KEYCARACTERISTIQUES =
              TENSION D'ALIMENTATION :        quantité V,
              SURFACE PUCE NUE :              quantité cm2,
              DIMENSIONS DISPOSITIF :         quantité cm x
                                              quantité cm x
                                              quantité cm,
              POIDS :                         quantité kg,
              TEMPERATURE DE FONCTIONNEMENT : quantité degrés,
              PRECISION :                     quantité unités,
              PERFORMANCES :                  quantité unités,
              COUTS :                         quantité dollars,
              EVOLUTIONS :                    quantité années>
```

Matériaux et Technologies Clés

- silicium micro usiné
- CCD
- technologie du silicium
- fibres optiques
- lasers
- effet Hall
- techniques de la micro ingénierie
- assemblage des multi puces
- effet piézoélectrique
- effet micro pyrotechnique

- silicium nu
- effet hygroscopique
- technologie de surface
- système sur isolateur
- technologie CMOS ASIC
- technologie de couches minces
- effet électrochimique
- chromatographie des gaz
- technologie GaAs
- technologie optique
- électronique de puissance
- technologie analogique
- technologie RF
- mémoires DRAM
- EEPROMS
- FPGA
- cellules standard
- PGA/PLD

Caractéristiques Clés

- tension d'alimentation
- taille de la puce
- dimensions
- poids
- température d'opération

- linéarité
- précision
- temps de réponse / vitesse
- coût
- évolution technologique / disponibilité
- nombre de transistors
- compatibilité électromagnétique
- consommation de puissance électrique

2.4.6 Exemples de blocs de la bibliothèque

Capteur de pression piézoélectrique

Le bloc correspondant est codé de la façon suivante :

```
<BLOCK capteur_de_pression IS organe mesurer MODEL = sensorp.vhd
            KEYCARACTERISTICS =
      <SERVICE interface_physique>
          <SIGNALS_P v,i : électrique><\SIGNALS_P>
            <SIGNALS_R P : fluidique><\SIGNALS_R>
          <CONSTRAINT v <= vcc, p <= pmax><\CONSTRAINT>
      <\SERVICE>
<\BLOCK>
```

Fichier sensorp.vhd

```
            SUBTYPE tension IS real ;
            SUBTYPE curent IS real ;
            SUBTYPE pression IS real ;
            SUBTYPE débit IS real ;

            NATURE électrique IS tension ACROSS curent THROUGH ;
            ALIAS terre IS electrique'reference ;
```

```
NATURE fluidique IS pression ACROSS débit THROUGH ;
ALIAS terre_fluidique IS fluidique'reference ;

ENTITY capteur_de_pression IS
            GENERIC (   h : real :=17.0e-6 ;
                        a : real :=1.0e-3 ;
                        r0 : real ;
                        rs : real) ;
            PORT (TERMINAL fp : fluidique ;
                  TERMINAL ep : électrique) ;
END ENTITY capteur_de_pression ;

ARCHITECTURE première OF capteur_de_pression IS
     CONSTANT e0 : real := 146.9e9 ;
     CONSTANT v0 : real :=0.1846 ;
     CONSTANT df : real :=e0*h**3/(12.0*(1.0sqr(v0));
     QUANTITY v ACROSS i THROUGH ep TO terre ;
     QUANTITY p ACROSS fp TO terre_fluidique ;
     QUANTITY w11 :real ;
  BEGIN
  (w11/h)**3 + 0.2522*w11/h == 0.000133*p*a**4/(df*h);
   i == v/(r0 + rs*2.5223*1.5895e9*w11) ;
END ARCHITECTURE première ;
```

2.4.7 Gestion du Temps

Les architectures décrites sont à fonctionnement *auto-temporisé*. Le comportement auto-temporisé assure que tous les événements arrivent au moment précis, mais pas à un instant spécifique. Une architecture auto temporisée est un bloc auto temporisé ou un ensemble de blocs auto temporisés interconnectés correctement.

Du point de vu du temps, une architecture HiLeS est une interconnexion de parties (blocs). Le temps et la séquence d'événements sont spécifiés à l'intérieur des blocs. Un bloc exécute une suite de calculs, dont l'initiation est causée par les signaux d'entrée et dont l'achèvement est indiqué par des signaux à événements à la sortie du bloc. Le séquencement des calculs est déterminé par la façon dont

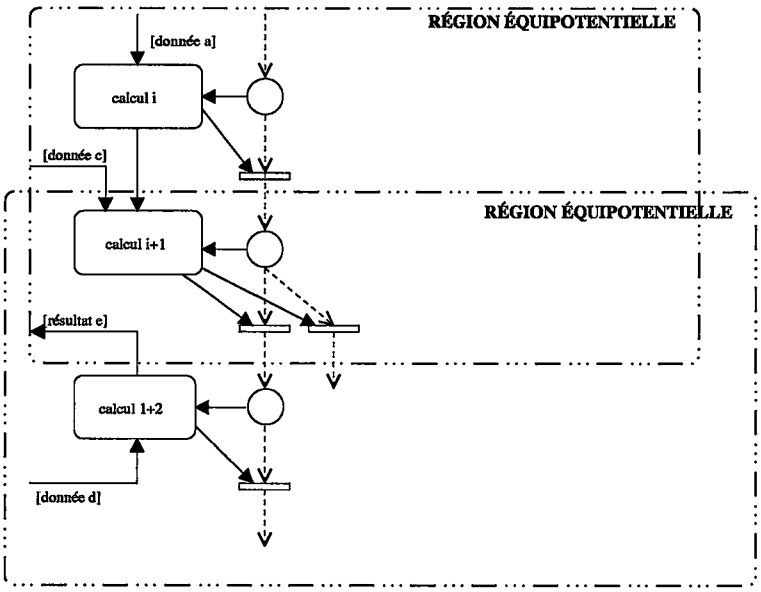

FIG. 2.12 – *Recouvrement des régions équipotentielles*

les blocs fonctionnels sont interconnectés. Le temps de calcul est déterminé par les retards imposés par les blocs fonctionnels entre l'initiation et l'achèvement et par les retards des interconnexions. La séquence des calculs est spécifiée à l'aide des réseaux de Petri interprétés proposes par HiLeS. Le réseau impose les relations temporelles de précédence, de concurrence et de simultanéité.

Etant donné une correspondance directe entre les descriptions architecturales proposées et le silicium, les relations d'occurrence des événements à différents endroits d'une puce nue, peuvent être interprétées différemment par des blocs à différents points. Le langage HiLeS permet de traiter chaque pas de calcul comme une région *équipotentielle*, c'est à dire tous les signaux à l'intérieur d'un pas de calcul sont considérés comme identiques à tous les points d'une interconnexion.

La signification physique des ces relations temporelles est illustrée par la figure 2.12. Cette caractéristique des architectures décrites assure que tout bloc est couvert par un ensemble de régions équipotentielles. Tout calcul génère une région équipotentielle à son entourage.

Par ailleurs, les contraintes à respecter aux terminaux d'un bloc fonctionnel, soient-elles fonctionnelles (comportements) ou de domaine (contraintes sur le séquencement correct des entrées) sont satisfaites par le type de signalisation des systèmes auto temporisés. La signalisation est tel que entre deux blocs qui

communiquent, il existe au moins une boucle fermée d'appel - réponse.

Etant donné qu'à l'intérieur d'une région équipotentielle les retards des interconnexions sont négligeables, les relations temporelles sont toujours respectées. Les connexions entre régions équipotentielles peuvent avoir des retards arbitraires et pourtant la signalisation doit être du type non sensitive aux retards ("delay-insensitive"). Nous utilisons la signalisation à quatre phases à retour à zéro. Cette signalisation assure une connexion correcte des blocs auto temporisés.

2.5 Description de la Méthode HiLeS

Nous allons décrire la méthode HiLeS en faisant référence aux figures 2.1 et 2.2.

2.5.1 Acquisition des Exigences, Besoins et Contraintes

Cet effort technique commence par l'identification, le rassemblement, et la définition des exigences de l'acquéreur et d'autres dépositaires d'enjeu sur le système à concevoir (les utilisateurs finaux du système, les fournisseurs du problème à résoudre, les fabricants, les mainteneurs, etc). Une liste des exigences essentielles et de la solution désirée doit être collectée, identifiée, clarifiée et mise en priorité. Cette phase est la première du processus de définition des exigences [47]. Les exigences, les besoins et les contraintes sont les objectifs, les conditions et les restrictions que le Micro Système conçu doit satisfaire.

Les données qui doivent être collectées à cette phase sont les suivantes:

- Concept d'opération: spécification des différentes utilisations de chaque Micro Système et du réseau de Micro Systèmes.
- Exigences fonctionnelles: spécification des fonctions que chaque Micro Système et le réseau de Micro Systèmes doit accomplir
- Les conditions d'exécution: celles ci fixent la façon dont chaque fonction doit être accomplie
- Les environnements normaux et induits dans lesquels les Micro Systèmes doivent fonctionner ou être utilisés
- Les contraintes de conception telles que l'utilisation ou non d'éléments réutilisables

- Les conditions concernant la disponibilité, l'interopérabilité, l'entretien, la fiabilité, la sûreté et la sécurité
- Les mesures d'efficacité qui reflètent les taux de satisfaction des exigences fixées par le cahier des charges
- Les contraintes concernant le développement, la production, l'essai, l'installation, la formation, le support/maintenance, et le démantèlement

D'autres exigences qui peuvent contraindre la conception du système doivent être identifiées, rassemblées et définies. Ceci inclut des données telles que:

- les plans du projet
- l'affectation d'équipes et l'organisation
- les métriques exigées
- les outils CAO exigés
- la base technologique disponible
- les ressources physiques et financières
- les standards nationaux, internationaux et industriels
- les lois, les régulations, les politiques, les procédures et les guides
- les décisions issues de la gestion

Ces exigences sont validées par la résolution de tous les conflits et des incohérences.

La deuxième phase du processus de définition des exigences consiste à développer l'ensemble des exigences techniques (cahier des charges) à partir des exigences issues de la phase précédente. Les tâches à réaliser pendant cette phase sont les suivantes :

- Etablir les priorités, les règles, les entrées, les sorties, les états, les modes et les configurations qui affecteront la phase de définition des exigences techniques
- Définir les exigences opérationnelles pour les utilisations prévues du système :
  - environnements d'utilisation et facteurs, normaux ou induits, susceptibles d'influer sur le fonctionnement du système

  - événements auxquels le système doit répondre
  - interfaces physiques et fonctionnelles (mécaniques, électriques, thermiques, données, et procédurales) y compris interactions physiques (forme et adaptation), frontière du système et interaction du système et son environnement (flux d'information et comportement)
  - ce que le système doit être capable d'accomplir (exigences fonctionnelles) pour satisfaire les exigences identifiées. Ceci inclut des facteurs tels que : testabilité, portabilité, disponibilité, fiabilité, maintenance, sécurité, et sûreté de fonctionnement
  - combien de fois le système sera utilisé, durée de cycle entre les utilisations, et combien de fois chaque fonction du système sera accomplie.
- Définir des exigences de performance :
  - les espérances de performance pour chaque condition fonctionnelle : temps d'exécution, nombre d'opérations par unité de temps
  - l'ensemble de mesures de performance : vitesse, consommation d'énergie, taille, poids, coût, EMI (Interférence Electromagnétique)
  - les indicateurs critiques de performance, dont le manque de satisfaction, pourraient faire courir au projet un risque économique, d'exécution ou de performance
  - l'approche de testabilité (fonctionnelle et de performance) pour chaque exigence
- Analyser les exigences du fournisseur du problème à résoudre pour :
  - définir les effets des facteurs humains,
  - établir les capacités et la temporisation,
  - définir les contraintes de conception,
  - définir les exigences de réalisation et de mise en service,
  - identifier les conflits et,
  - déterminer les critères pour la résolution de conflits
- identifier les exigences qui posent problème
- faire le bilan de l'ensemble des exigences techniques acceptables et assurer leurs consistances et leurs complétudes.

2.5.2 Génération des Descriptions Architecturales

La génération des descriptions architecturales est un procédé de définition de solutions employées pour concevoir une solution acceptable. Au niveau le plus haut d'abstraction cette description correspond à une esquisse architecturale servant à valider les exigences acquises dans la phase d'acquisition des exigences. Aux niveaux suivants, une description architecturale sert à valider les spécifications issues du niveau supérieur. Cette description donne lieu à une *architecture générique* indépendante de la technologie de fabrication.

Ainsi, la solution obtenue doit satisfaire:

- les exigences techniques du système résultant de leur définition même
- les exigences techniques additionnelles que ce processus de définition de solutions peut lui même engendrer.

Alors, les exigences, les besoins, les contraintes et les spécifications sont validés.

Formalisme de représentation de l'architecture générique

La représentation d'une architecture est faite à l'aide du modèle graphique HiLeS présenté auparavant. Le formalisme suggéré suit une approche d'analyse structurée et de modélisation de l'information. Cette définition abstraite de la solution prend la forme d'un *flot* de matière, d'énergie et d'information entre *blocs* structurels. Ce flot est modélisé par des *canaux* qui transportent des signaux de natures physiques diverses (électrique, thermique, électromagnétique, etc.). Les blocs encapsulent des parties des exigences techniques et sont caractérisés par des *services* (comportements, fonctions et opérations) fournis / demandés et par des attributs (grandeurs, caractéristiques et données) encapsulés. La modélisation des *services* d'un bloc est faite par un *graphe de contrôle* et un *graphe de flux de données*, favorisant une réalisation *asynchrone*(partition contrôle / données).

Établissement de l'ensemble de représentations logiques de la solution

Le format sélectionné pour définir les différentes solutions permet une meilleure utilisation des technologies potentiellement disponibles. Les activités à réaliser sont :

Sélection d'une division appropriée en sous-structures : L'objectif de cette activité est de sélectionner une division d'objets et de composants que nous allons appeler sous-structures. Les exigences techniques du système sont assignées à ces sous-structures d'une façon appropriée. Il est possible que certaines exigences ne soient pas prises en compte à la fin de cette phase. Pour guider le concepteur dans cette activité nous proposons de suivre une approche en quatre étapes (voir figure 2.2).

Partition par flots de matière, d'énergie et d'information Le Micro Système interagit avec l'environnement par des flots de matière ou d'énergie. Il les capte pour les mesurer ou les transformer. Il peut aussi les produire pour communiquer ou comme résultat des transformations en amont. Ces flots peuvent aussi servir à transporter de l'information. Cette première partition va permettre d'identifier les différents domaines disciplinaires et les besoins d'interfaçage et de séparer les aspects informationnels.

Les interfaces peuvent avoir une taille plus grande que les composants actifs par le fait des contraintes de chaque domaine disciplinaire : les capteurs optiques sont contraints par la taille des lentilles, les capteurs chimiques sont contraints par la taille des filtres, les actionneurs fluidiques sont contraints par la taille des débits, le boîtier du Micro Système est contraint par son environnement de travail (milieu biologique, chimique, etc.). Il est pourtant nécessaire à ce stade d'identifier et de définir les interfaces entre domaines multidisciplinaires : électrothermique, optoélectronique, électrochimique, électromagnétique et électromécanique (y compris l'interfaçage fluidique). Etant donné que la base technologique des Micro Systèmes est issue de la microélectronique, une attention spécifique doit être fournie aux interfaces électriques.

Cette partition donne lieu à de nouvelles spécifications et permet la définition de la structure externe du Micro Système en terme de fonctions génériques 2.4.5. Les activités à réaliser sont :

- définition des interfaces électriques, fluidiques, thermiques, optiques, électromagnétiques, chimiques et mécaniques (fonctions interconnecter, ingérer, expulser, recevoir, émettre, convertir, distribuer);

- définition de l'interfaçage entre le Micro Système et son environnement de travail (fonction empaqueter);

- définition des canaux d'entrée et de sortie et de leur nature matérielle et énergétique (y compris les signaux et leur type)

- définition des supports physiques (fonction supporter);

Partition Fonctionnelle Cette phase utilise les fonctions génériques proposées en 2.4.5. Cette activité produit la définition de l'architecture interne du Micro Système. Nous proposons une partition par groupes fonctionnels. Ces groupes ont été choisis de manière à constituer des blocs de base de la bibliothèque de blocs:

- Définition des blocs capter, actionner et conditionner;

- Définition des blocs superviser en utilisant les fonctions génériques associer, décider, temporiser, mémoriser, coder et décoder;

- Définition du bloc communiquer en utilisant les fonctions génériques associer, décider, temporiser, mémoriser, coder et décoder;

- Définition du bloc alimenter;

Partition Discret / Continu Cette phase consiste à définir le comportement ou les réponses aux stimuli pour chaque bloc identifié dans la phase antérieure en vue d'une modélisation en temps continu ou en temps discret. Le type de modèle et sa complexité dépendent étroitement du domaine disciplinaire et du niveau de détail désiré. Le comportement des blocs qui traitent l'énergie et la matière peuvent être décrits par des modèles de temps continu avec des signaux continus (équations différentielles). Le comportement des blocs qui traitent l'information est décrit en utilisant le modèle proposé par HiLeS: séparation entre contrôle et traitement de l'information.

Ainsi, pour les blocs qui traitent la matière et l'énergie, le concepteur doit identifier et définir le type de modélisation à utiliser. Les choix sont par exemple:

- un ensemble d'équations différentielles décrivant la relation de causalité entre les stimuli et les réponses;

- les fonctions de transfert du bloc (à paramètres repartis ou concentrés);

- une modélisation basée sur l'analogie aux réseaux électriques;

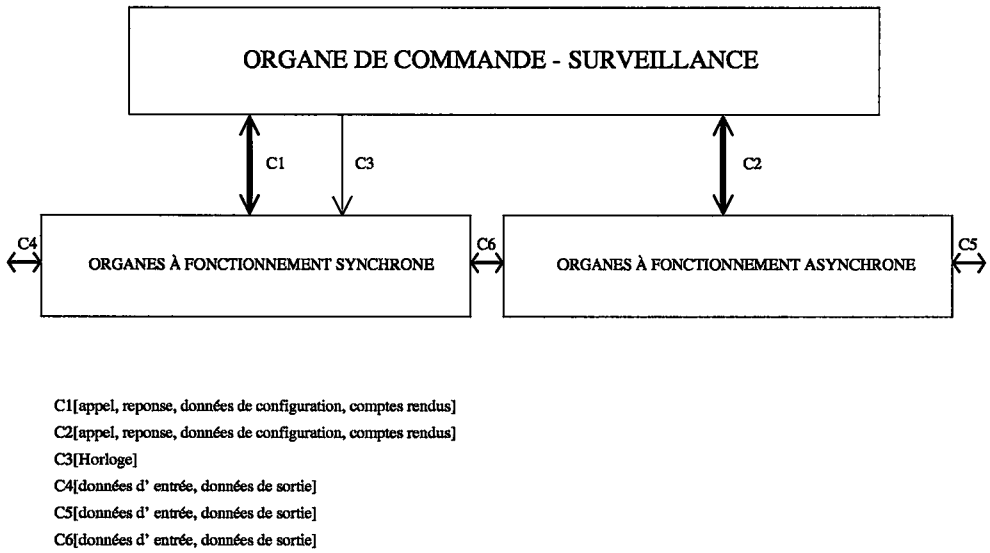

FIG. 2.13 – *Structure Générique du Système de Commande d'un Micro Système*

Partition contrôle / données Pour les blocs qui traitent l'information, le concepteur doit identifier les flots de traitement d'information et le contrôle du traitement par flux de jetons. L'outil fournit par HiLeS est le réseau de Petri interprété (réseau de Petri accompagné d'un graphe de traitement de l'information). Le graphe de traitement de l'information montre les séquences de transformation à l'aide des blocs fonctionnels. Cette structure correspond à l'exécution d'un algorithme par commande distribuée. Pour guider le concepteur nous allons utiliser la structure générique de commande d'un Micro Système proposée par la figure 2.13. Tout organe peut avoir un fonctionnement synchrone ou asynchrone. Les organes peuvent échanger de l'information entre eux. La communication entre le bloc de commande - surveillance s'est fait par appels, réponses, comptes rendus, horloges et données.

Le comportement d'un organe est décrit par le réseau de Petri montré par la figure 2.14.

Le concepteur doit identifier :

- l'algorithme et la vérification à implémenter,
- les attributs d'interface, par exemple le protocole, la réponse aux événements, le changement d'état ou de mode et le flot de données,
- les états initiaux et finaux,

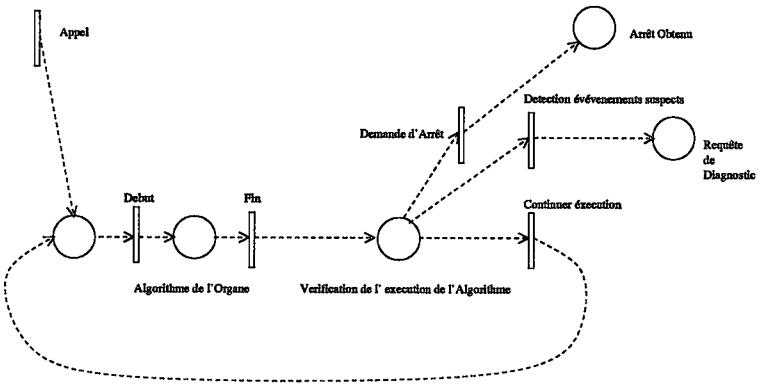

FIG. 2.14 – *Structure Générique d'un organe*

- les états et les modes de chaque sous-calcul,
- les contraintes temporelles et les temps d'exécution pour l'ordonnancement des opérations, et
- les modes de panne et leurs effets.

L'algorithme d'un organe est spécifié par des pas de calcul (instructions) et son séquencement.

Chaque pas, correspondant à une modification du comportement de l'algorithme, exprime, après que l'on ait spécifié les d'entrées et de sorties:

- les événements d'entrée associés qui vont impliquer une nouvelle étape dans l'évolution de l'algorithme,
- les pre conditions qui assurent la prise en considération de ses événements et qui tiennent compte des évolutions passées de l'algorithme, {itemles post conditions induites par ces événements,
- les evenements conséquentes sensibilisées ou évolutions futures susceptibles d'intervenir dès la prise en compte des événements d'entrée.

Exigences techniques dérivées

Cette phase consiste à:

- montrer les exigences associées à des sous structures
- faire l'expansion des exigences techniques à un niveau de détail plus fin

- valider les exigences techniques dérivées tant individuellement que comme un ensemble, par exemple:
  - faisabilité
  - l'absence d'ambiguïté
  - testabilité
  - vérifiabilité
  - absence de redondance
  - connectivité
- assurer par validation et par suppression de conflits que les représentations des solutions sont en accord avec les ensembles correspondant aux exigences techniques

2.5.3 Vérification Structurelle

La vérification est une tâche primordiale dans le processus entier de conception. Elle se focalise plus particulièrement sur l'intégration et le test des blocs numériques et analogiques, où plus d'un tiers du cycle de développement y est dépensé. La vérification vise à déterminer en même temps le caractère correct d'un ou plusieurs modèle(s) d'un système hétérogène. Le modèle peut être à différents niveaux d'abstraction. La validation peut être basée soit sur la simulation, la vérification formelle, ou l'émulation, soit sur un prototype réel. Ici, nous nous limiterons à la vérification du prototype virtuel. La simulation est le principal outil de validation des systèmes embarqués, mais l'émergence de la vérification formelle de propriétés offre une solution complémentaire. Dans le cas d'un système complet sur silicium, l'émulation permet de s'affranchir des limites en performances de la simulation pour valider le prototype dans des temps non prohibitifs.

Nous proposons de faire une vérification formelle étant donné que la description comportementale utilise les réseaux de Petri Interpretés.

On peut distinguer deux types de vérification formelle:

- La vérification de la spécification; Il s'agit de vérifier une propriété abstraite d'un modèle fonctionnel de haut niveau.

- La vérification de l'implémentation; il s'agit de vérifier si un modèle d'un bas niveau implémente correctement un modèle de plus haut niveau et/ou satisfait une propriété dépendante de l'implémentation.

La vérification de l'implémentation sur du matériel asynchrone et dédié est un domaine relativement bien développé (voir Chapitre 1).

Les propriétés abstraites à vérifier sont de deux types pour une description comportementale quelconque:

- Les propriétés de sécurité, qui stipulent que le système ne passera jamais dans une configuration indésirable (inter blocage, émission de sorties non désirées...), et ce quelles que soient les entrées appliquées et la résolution d'éventuels choix non déterministes à l'intérieur du modèle du système.

- Les propriétés d'existence, qui stipulent qu'une configuration désirée se produira éventuellement.

Des études structurelles sur les réseaux de Petri ont été effectuées. Elles permettent de s'assurer qu'un réseau possède ces propriétés de sécurité et de vivacité. Cette vérification permet de déceler les comportements indésirables ayant pour origine une structure mal formée.

Le principal avantage de l'approche théorique par réseaux de Petri, pour la vérification structurelle, est qu'elle permet des abstractions conservatives pour diminuer l'espace des états à rechercher. L'abstraction est la technique qui permet de faire face à l'explosion de l'espace des états. Le modèle est remplacé par des approximations successives conservatives, i.e. qui interdisent que la propriété devienne satisfaite alors qu'elle ne l'était pas dans le modèle original.

Par ailleurs, il est important de déterminer les mauvais fonctionnements provenant des entrées sorties. La principale cause de mauvais fonctionnement est le caractère parallèle de systèmes décrits. On doit identifier des comportements indésirables dus à des séquences isolées, à des séquences simultanées ou parallèles et au partage des ressources. Cette étude de fonctionnements défectueux sera toujours topologique, donc très liée au circuit définitif. Elle s'applique sur la représentation initiale et sur les représentations locales, ce qui la rend robuste à manipuler sur des structures complexes.

Nous proposons des procédures de simulation pour l'activité de dépistage des comportements indésirables mentionnés auparavant. Plusieurs outils de simulation de Réseaux de Petri sont disponibles sur le marché, à l'intérieur de la

communauté scientifique et sur Internet pour aider le concepteur à cette activité. Ce choix est guidé par le souci d'utiliser des outils disponibles commercialement et par leur acceptation de la part des industriels.

Par ailleurs, l'utilisation d'une architecture générique de Micro Système permet au concepteur d'assurer dès le début du processus de conception l'utilisation des blocs intégrés, déjà vérifiés (ayant une structure bien formée).

2.5.4 Description du Domaine Economique

Pour prendre une décision d'investissement, il faut évaluer les incidences économiques sur l'ensemble d'un projet de développement d'une application basée Micro Systèmes. Le concepteur doit être capable de choisir l'architecture la plus rentable. Nous proposons un modèle financier, appelé domaine économique. Un modèle financier est ajouté à chaque bloc de la librairie. Ce domaine décrit le modèle choisi de valorisation du projet. Plusieurs modèles sont disponibles :

- coût global,
- délai de remboursement,
- valeur actuelle nette,
- taux interne de rentabilité,
- étude de sensibilité.

Ce modèle inclut le fait que le composant soit disponible ou non, et le fait qu'il n'existe pas ou qu'il soit en cours de développement.

Nous proposons d'ajouter deux caractéristiques clefs appelées : date de disponibilité et statut technologique. À partir de ces dernières, le modèle économique d'un composant peut être pris en compte dans le domaine économique de l'architecture dont il fait partie.

2.5.5 Validation par Simulation Mixte

La simulation représente une difficulté car les systèmes sont hétérogènes. Ils contiennent des composants numériques, analogiques et logiciels qui doivent être simulés en même temps pour s'assurer du bon fonctionnement du système complet (simulation mixte). Elle permet de vérifier, avant la fabrication d'un prototype matériel, les interactions hétérogènes du prototype virtuel, afin d'une

part de paralléliser les développements des différents composants, et d'autre part d'augmenter la fiabilité de la vérification fournissant une méthode redondante pour dépister les erreurs de conception. Pour résoudre ce problème le standard IEEE VHDL-AMS a été proposé en 1999. Nous avons adopté ce standard comme interface de communication de modèles et de descriptions entre la conception amont et la conception aval. La méthode proposée, néanmoins, est indépendante de l'interface choisi avec la conception aval. La méthode cherche à être multi modèle et multi langages. Les difficultés de ce type de simulation proviennent des différences de performance exigées et réalisées entre un simulateur à événements discrets, un simulateur à temps continu et le temps réel. Nous avons adopté le standard IEEE VHDL-AMS comme outil pour la phase de validation par co-simulation. Nous proposons une passerelle entre la conception de haut niveau de Micro Systèmes et la modélisation virtuelle par VHDL-AMS, par l'application de la méthodologie HiLeS. Etant donné que les simulations d'un prototype virtuel tournent à une vitesse de plusieurs degrés de magnitude moins vite que le temps réel, le concepteur doit faire un compromis entre le niveau de précision temporelle exigé pour vérifier les interactions hétérogènes, et la vitesse de simulation exigée pour exécuter le prototype virtuel. La méthode permet la description du comportement d'un composant à différents niveaux d'abstraction et cette composition permet d'avoir un contrôle du temps d'exécution de la simulation.

2.5.6 Représentation physique de l'Architecture Générique (choix technologiques)

Un système étant constitué d'un ensemble de sous structures, sa représentation physique sera déterminée par l'ensemble des représentations physiques de ses sous structures. A chaque sous structure doit être associé un *bloc* (fonctionnel ou structurel : voir définition en 2.3.3) selon les différentes technologies ou natures physiques. Ces blocs sont classés dans une bibliothèque suivant les attributs de classification décrits dans 2.4.5 et présentés sous la forme de matrices qui contiennent aussi potentiellement toutes leurs combinaisons possibles. Chaque combinaison trouvée est une proposition de composition d'une architecture (voir figure 2.15). Le concepteur doit produire l'architecture correspondante en assemblant les blocs proposés en suivant le flux de matière, d'énergie, et d'information. Chaque alternative d'architecture doit être décrite et un acronyme

doit être fourni.

A ce stade de la conception, nous disposons d'une architecture du système dont le comportement fonctionnel a été validé. Cette architecture a été établie indépendamment des technologies (matériaux et matériels) nécessaires à sa mise en oeuvre. A partir de cette architecture générique, des choix technologiques vont être établis et vont permettre d'obtenir plusieurs variantes architecturales. Ces variantes vont être évaluées selon des critères particuliers (performances, coûts, consommation, etc.) pour finalement ne retenir qu'une seule architecture. Les principaux choix technologiques seront relatifs :

- aux matériaux et technologies clés pour l'ensemble des constituants,
- pour la partie électronique : partitionnement analogique / numérique, materiel / logiciel, ASIC / FPGA / Microprocesseur, circuits synchrones / asynchrones,
- pour l'interface électronique / partie sensorielle : intégration monolithique / assemblage hybride, etc.,
- pour l'interface microsystème / environnement immédiat : packaging, interconnexion (PGA, BGA, DIP, etc.).

Du point de vue de l'outil informatique, le processus se déroule de la façon suivante :

À partir du cahier des charges, le concepteur va sélectionner par l'intermédiaire d'une première interface spécifique les composants et sous-systèmes pouvant assurer les diverses fonctions au sein du microsystème. Ces éléments sont contenus dans la partie correspondante de la base de données.

Les différents champs de cette interface sont :

- la Nature du composant : mécanique, thermique, fluidique, optique, électrique, électromagnétique.
- la Fonction : capter, traiter, décider, mémoriser, communiquer, actionner, polariser, alimenter, protéger (packaging), dater (temporiser).
- les Mots Clés : identification, domaine d'application, technologies et matériaux clés ; limité à cinq mots clés.
- les Caractéristiques : données techniques et financières caractérisant le composant.

PARTITION STRUCTUREL (SOLUTION LOGIQUE)	OPTIONS DE SOLUTIONS PHYSIQUES (BLOCS DE LA BIBLIOTEQUE)				
Sous Structure 1	Bloc11	Bloc12	Bloc13	Bloc14	
Sous Structure 2	Bloc21	Bloc 22			
Sous Structure 3	Bloc31				
Sous Structure 4	Bloc41	Bloc42	Bloc43	Bloc44	Bloc45
...	...	...	...	...	...

ARCHITECTURE 1 : Bloc11 + Bloc21 + Bloc31 + Bloc41 ...
ARCHITECTURE 2 : Bloc11 + Bloc22 + Bloc31 + Bloc42 ...
ARCHITECTURE 3 : Bloc13 + Bloc21 + Bloc31 + Bloc44 ...
...

FIG. 2.15 – *Assignation représentation logique - représentation physique*

– Type de Modèle: à temps continu ou à temps discret.

Ainsi, dès que le concepteur aura rempli ces champs avec les informations adéquates, le système gérant la base de données proposera pour chaque composant ou sous-système des solutions différentes sous la forme d'un tableau répondant au cahier des charges imposé via l'interface.

2.5.7 Sélection de la meilleure architecture

Pour la sélection des meilleures solutions, les critères principaux de sélection ont été identifiés dans la phase de définition des exigences de performance.

Du point de vue de l'outil informatique, une deuxième interface avec le concepteur doit être proposée. Les différents champs de cette interface sont:

– critère

– pondération

La figure 2.16 montre la structure du processus de sélection de la meilleure alternative d'architecture.

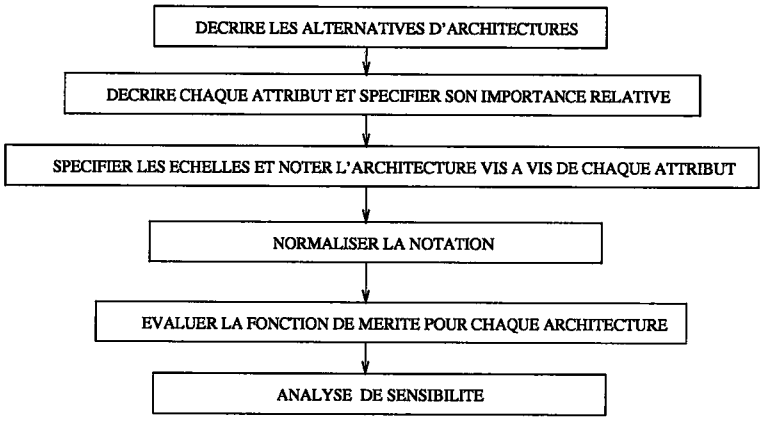

FIG. 2.16 – *Processus de sélection*

Attributs d'une Architecture

Un attribut représente une qualité du fonctionnement de l'architecture. Sa mesure est un critère d'évaluation (exigence de performance) en conséquence il doit être quantifiable. Un attribut non considéré ici, n'influencera pas le processus de sélection. L'importance relative Ij de chaque attribut Ai est établie par des méthodes de comparaison et de rangement. Les critères spécifiques sont alors rangés par ordre d'importance et un poids normalisé est assigné pour montrer l'importance de chaque critère spécifique.

Importance relative des attributs

Différentes échelles (taux, intervalle, ordinale ou composée) sont employées pour établir l'importance relative entre critères et pour fournir les moyens de mesurer les préférences. Le choix d'un type particulier d'échelle pour modéliser un attribut dépend de la nature de l'information disponible.

L'échelle à taux est utilisée pour mesurer des quantités physiques (consommation de puissance, vitesse, coût, poids, surface, etc.). Elle spécifie la valeur maximale Aj max et la valeur minimale Aj min pour l'attribut j. La valeur maximale correspond à la performance la plus désirable et la valeur minimale correspond à la performance qui est acceptable.

L'échelle ordinale est utilisée pour modéliser un attribut qui peut être seulement qualitatif.

L'échelle d'intervalles est utilisée pour des attributs pour lesquels il existe

CRITERES DE SELECTION		ARCHITECTURES			
	IMPORTANCE RELATIVE	A1	A2	A3	A4
ATTRIBUT 1	0.4	0	-1	-1	1
ATTRIBUT 2	0.3	0	-1	-1	1
ATTRIBUT 3	0.1	0	0	-1	1
ATTRIBUT 4	0.1	0	1	1	0
ATTRIBUT 5	0.1	0	1	0	1
EVALUATION		0	-0.5	-0.7	0.9
EVALUATION NORMALISÈE		0.44	0.12	0	1

FIG. 2.17 – *Evaluation des alternatives d'architecture*

seulement de l'information quantifiable. Elle est utilisée de deux manières : pour modéliser des attributs dont zéro est la valeur relative et pour transformer une échelle ordinale en une échelle d'intervalle numérique.

L'échelle composée est utilisée pour des attributs qui sont générés à partir de calculs ou composés à partir d'autres attributs.

Chaque alternative est alors évaluée pour chaque attribut et une valeur de fonction de mérite peut être obtenue afin de choisir la meilleure solution architecturale (voir figure 2.17).

2.6 Conclusions

L'approche descendante proposée permet une nouvelle conception des Micro Systèmes partant des exigences et pas de la microélectronique, tel que cela est courant aujourd'hui.

La partition fonctionnelle proposée de l'architecture interne d'un Micro Système offre un cadre approprié pour la modularité et pour la conception aval.

HiLeS a amené une conception modulaire qui permet le passage direct du réseau de PETRI représentant le fonctionnement du système désiré à une réalisation physique indépendante de la vitesse.

La description hiérarchique par réseaux de Petri permet une spécification et une validation locales et pas seulement globales, ce qui rend utile la méthode pour le traitement des structures complexes. Vérifier les bonnes propriétés de façon locale ne garantit pas une optimalité stricte, par contre, elle rend possible

le traitement de systèmes complexes très rapidement et en limitant les difficultés.

Chapitre 3

INTRODUCTION À L'AUTONOMIE EMBARQUÈE

3.1 Introduction

Jusqu'à présent, nous avons présenté la méthode de Conception Architecturale de Micro Systèmes dans un contexte exempt de défaillances : tout compte rendu retourné par un bloc traduit une fin normale d'exécution d'une activité de commande. Toutefois, lorsque le Micro Système est fabriqué, en cours d'opération et pendant son vieillissement, un fonctionnement fiable n'est pas toujours assuré.

Des analyses de la physique de la défaillance seront menées dans le cadre de la conception aval [10]. Cette analyse identifie, à l'intérieur du micro système, les endroits de défaillance potentielle, le mécanisme de défaillance (électrique, mécanique, chimique, etc.) et les modes de défaillance (court circuits, circuits ouverts ou déviations des paramètres hors tolérances). La modélisation de ces défaillances par des modèles adéquats, permettra le calcul de la durée moyenne avant défaillance. Si ce paramètre est plus petit que la durée de vie utile du Micro Système, alors des études de sensibilité devront être entreprises pour augmenter cette durée au niveau demandé par les spécifications, par modification des paramètres de conception.

Dans le contexte de la conception amont (étant donné la complexité des applications envisagées - réseaux de Micro Systèmes - et leurs exigences de fiabilité, de sûreté de fonctionnement et d'autonomie) l'adjonction d'un *système de test en temps réel de toute l'électronique embarquée* est indispensable. Le but d'un tel système étant la recherche des composants défaillants, la réussite de cette activité dépend fortement du degré de testabilité du système électronique.

Le domaine d'application choisi est la réalisation de circuits multi capteurs capables de mesurer des paramètres caractérisant l'usage de différentes parties d'un système opérationnel. Cette surveillance permettra de déterminer avec plus de précision les opérations de maintenance préventives à effectuer, et de servir de systèmes d'alarme afin d'assurer la sûreté de fonctionnement du circuit observé. Cette fonctionnalité doit répondre à des questions telles que :

- la mesure appartient-elle à un domaine de fonctionnement nominal (permis) ou est-elle en dehors?,
- si c'est un fonctionnement normal, est-il nécessaire de mémoriser cette dernière valeur? autrement dit, la variation par rapport à la précédente est-elle significative?
- si c'est un fonctionnement anormal, est-ce que les conditions environnementales ont brusquement changé ou alors y aurait-il une erreur au niveau de l'acquisition (problèmes de capteurs?) ou du traitement (circuits analogiques ou numériques?).

En outre, le système de surveillance doit être suffisamment autonome pour prendre les décisions qui s'imposent en cas d'événements anormaux. En effet, sachant qu'un système de test de l'ensemble des éléments composant le système (capteurs, niveaux d'énergie, chaînes d'acquisition et de traitement, interface de communication,...) est en place, il sera capable d'établir un diagnostic complet sur l'état général, de continuer à fonctionner même en cas de perte d'un ou plusieurs capteurs, de décider de changer le mode de fonctionnement pour optimiser la consommation d'énergie ou de s'arrêter si le niveau d'énergie est trop faible.

Partant de ces constats, nous allons proposer un organe superviseur intégré qui aura les fonctions suivantes :

- commande du Micro Système,

- surveillance du fonctionnement du Micro Système et
- gestion autonome de ses modes de fonctionnement.

Cet organe pourra être personnalisé à l'application, par l'utilisation des blocs génériques que nous allons proposer par la suite (détecter, diagnostiquer, tester, décider et programmer). Ces blocs pourront être imbriquées à la commande primaire pour surveiller son état et sa sûreté de fonctionnement.

3.2 Architecture de surveillance repartie temps réel

Nous avons utilisé le modèle de commande - surveillance développé au LAAS ([16], [18], [17]) pour pouvoir, par son adaptation, proposer une architecture générique d'un bloc superviseur intégré au Micro Système, répondant aux besoins exprimés ci-avant.

Lorsqu'une requête de commande est envoyée vers un bloc, rien ne garantit que le compte rendu reçu sera celui attendu. La déviation caractérise la défaillance (d'où la notion de normal/anormal). Deux raisons peuvent être à l'origine de ce dysfonctionnement : déficience d'un élément du bloc de traitement ou d'un élément du bloc de commande.

Dans notre approche, le modèle de contrôle primaire (celui qui ne peut être décomposé sous forme de sous structures) est tel que le réseau de commande est toujours imbriqué dans le bloc traitement (réseau de Petri Interprété), définissant de cette manière une architecture de commande répartie et modulaire à un seul niveau. Nous allons l'appeler *niveau de commande élémentaire.* Il est naturel d'imaginer, ensuite, une structure hiérarchisée de décision dans laquelle un bloc n'est lié qu'à un seul bloc du niveau supérieur, tandis qu'il coordonne plusieurs blocs du niveau inférieur.

Chaque niveau de cette hiérarchie se charge de mettre en oeuvre les ordres qui lui sont donnés par le niveau supérieur. Ceci en utilisant les possibilités qu'offre le niveau immédiatement inférieur tout en respectant les contraintes qui sont prises en compte à ce niveau de description. Par ailleurs, à l'intérieur d'un même niveau, un découpage en blocs indépendants, correspondant à des fonctions génériques, permet de mieux maîtriser la complexité de l'application. Cette structure génère la notion de *sous système commandé.* Chaque bloc de

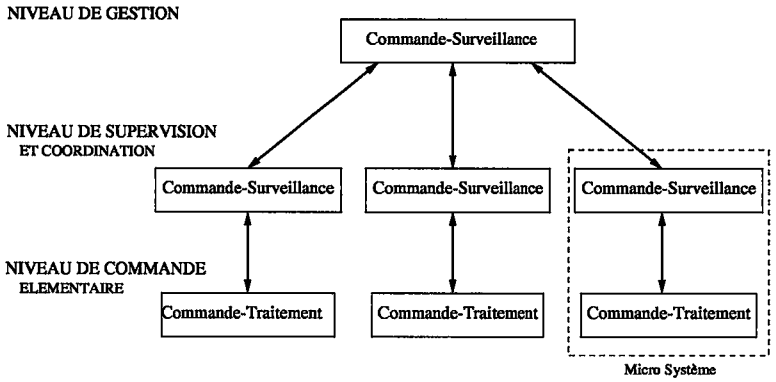

FIG. 3.1 – *Structure Hiérarchisée de Commande - Surveillance*

cette hiérarchie joue le rôle de racine vis à vis d'une partie de l'arborescence et les opérations qu'il offre à son supérieur sont réalisables par les blocs composant ce sous arbre de manière autonome. Cette partie de l'arborescence constitue le sous système commandé associé au bloc considéré.

De ce fait, chacun des blocs participant au fonctionnement de cet ensemble possède des éléments de connaissance que l'on ne retrouve dans aucun autre bloc. Plus précisément, dans une hiérarchie, lorsqu'un niveau reçoit une requête, il possède toutes les connaissances nécessaires à son exécution, mais il n'a pas, en général, la connaissance d'une solution à appliquer en cas de défaillance. Ceci est vrai si la défaillance n'a pas son origine dans le bloc où elle est détectée. Dans ce cas, effectivement, le bloc qui ne peut traiter la défaillance va en informer son niveau supérieur car il ne peut lui seul décider de remettre en cause la requête. Le traitement des situations d'exception doit être confié au niveau qui a généré la requête. C'est le cas d'un réseau de micro systèmes coordonné par une unité de contrôle éloignée.

Pour éviter qu'un seul niveau ait toutes les connaissances, chaque bloc a son propre système de prise en compte des défaillances ou bloc de surveillance *temps réel* (il doit donc agir sur le sous système commandé dans des délais compatibles avec sa dynamique). Nous aboutissons alors à la structure hiérarchisée de commande - surveillance montrée par la figure 3.1, dont nous définissons trois niveaux de commande hiérarchisée: le *niveau de commande élémentaire*, le *niveau de supervision - coordination* et le *niveau de gestion.*

Concernant les fonctions de *la commande élémentaire*, elles vont permettre d'analyser les changements des conditions d'environnement dans lesquelles évo-

luent avec le temps les systèmes surveillés afin d'améliorer:

- leurs fiabilités et par conséquent leurs durabilités, en détectant suffisamment tôt la détérioration et les pannes naissantes dans des mécanismes critiques,

- l'établissement d'un historique par mémorisation des pannes et des événements exceptionnels,

- la prise de mesures de corrections appropriées.

Ces fonctions ont pour but d'assurer le traitement (identification des mesures pertinentes, mémorisation de l'historique) des grandeurs physiques surveillées (accélération, température, décharges électrostatiques, etc.). Le Micro Système sera disposé dans des endroits stratégiques soumis à des contraintes d'environnement extrêmes et nécessitant une surveillance particulière (par exemple: les ailes ou le fuselage d'un avion). Ces fonctions seront assurées par des organes spécialisés.

Concernant les fonctions de *la supervision coordination*, elles vont assurer l'autonomie du Micro Système, la surveillance du niveau de contrôle élémentaire, la gestion des modes de fonctionnement, l'adaptation à l'évolution de l'environnement du Micro Système et la communication avec le niveau de gestion.

Le niveau de *gestion* correspond à un dispositif géographiquement éloigné et ses fonctions sont d'assurer le contrôle du plus grand nombre possible de Micro Systèmes (unité de contrôle éloignée). Cette unité peut à tout moment interrompre les activités du Micro Système en lui envoyant des commandes pour changer le mode de fonctionnement courant, générer des tests et renvoyer les résultats correspondants afin de vérifier le bon déroulement des opérations, ou encore obtenir des informations concernant l'environnement observé (historiques des mesures, des alarmes,...), etc.

Nous allons nous intéresser plus spécialement aux niveaux de commande élémentaire et de supervision coordination constituant le Micro Système, laissant la conception de l'unité de contrôle éloignée aux prospectives de notre travail.

3.3 Spécification d'un organe superviseur intégré

Le niveau de Supervision - Coordination se chargera de surveiller et de commander toute l'électronique opérationnelle d'un Micro Système (niveau de commande élémentaire) et de communiquer (envoi des comptes rendus, réception de requêtes) avec l'unité de contrôle éloignée. Notre objectif, dans cette section, est de définir l'architecture générique d'un Organe Superviseur Intégré appartenant au niveau de surveillance - coordination, et par la suite, nous allons nous concentrer dans la spécification de ce niveau.

3.3.1 Définition de l'organe superviseur integré

Le superviseur integré d'un micro système (niveau de surveillance - coordination) a comme fonctions le contrôle et la coordination de toutes les activités executées par les organes autres que le superviseur (niveau de commande élémentaire). Il a la capacité d'envoyer la demande de début d'activité à chaque organe coordoné, ainsi que la demande d'arrêt de l'organe quand le superviseur le considère nécessaire.

À son tour, chaque organe renvoie au superviseur un message indiquant son état. Ainsi, le superviseur est toujours au courant et il peut prendre les décisions necessaires en cas de présence d'un symptôme de defaillance du Micro Système.

Le superviseur integré est composé de deux blocs géneriques: le bloc de commande et le bloc de surveillance.

3.3.2 Communication Inter Niveaux

La communication entre le niveau de supervision - coordination et le niveau de commande élémentaire est purement orientée commande. La requête envoyée vers le niveau inférieur peut être modélisée par un jeton, en ayant mis à jour les informations nécessaires à son exécution (sélection du mode de fonctionnement), en imposant à l'organe de retourner le jeton sur deux conditions spécifiques:

- à la détection d'un fonctionnement suspect (anomalie) par les blocs de détection imbriqués dans l'organe;
- à la demande du niveau supérieur, en cas de changement de mode de fonctionnement ou comme compte rendu d'exécution.

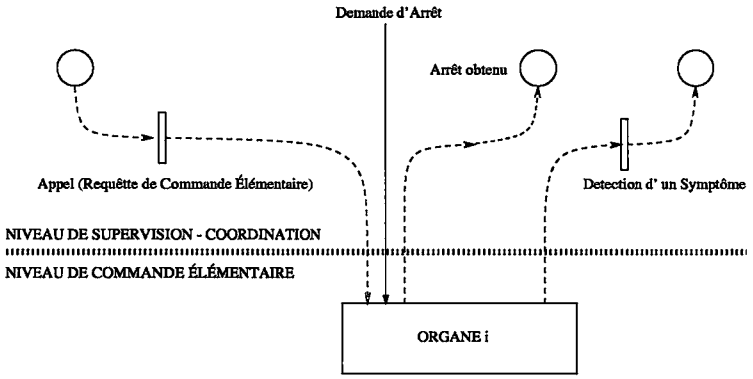

FIG. 3.2 – *Modèle de Communication Inter Niveaux*

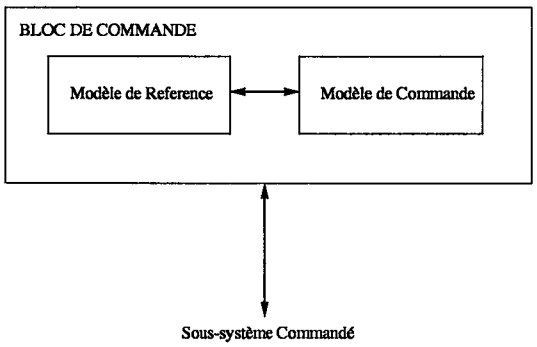

FIG. 3.3 – *Architecture de Commande à deux Modèles*

Le mécanisme employé est illustré par la figure 3.2.

3.3.3 Le bloc de Commande

Le bloc de commande est basé sur l'utilisation conjointe de deux modèles, le modèle de commande et le modèle de référence (modèle du niveau de commande élémentaire). Ces deux modèles s'appuient sur le concept d'activité et utilisent les réseaux de Petri interprétés (voir figure 3.3).

Dans chacun de ces deux modèles, des contraintes distinctes sont représentées :

- Dans le modèle de commande sont représentées toutes les contraintes liées au traitement donné par l'organe commandé. Il s'agit donc de contraintes opérationnelles ;

- Dans le modèle de référence sont représentées toutes les contraintes liées aux organes alloués et aux ressources allouées à chaque organe (gestion, redondance, séquences obligatoires etc.). Il s'agit donc de contraintes structurelles et matérielles.

Modèle de Référence

Le modèle de référence, pour un bloc du niveau supervision coordination, représente toutes les activités (et les contraintes qui les lient) réalisables par l'organe sous son contrôle. Il est important de noter que cette modélisation ne tient aucun compte d'une commande particulière. Cela garantit un degré de couverture important des services offerts par l'organe.

Ainsi, toutes les contraintes liées à la structure physique du Micro Système sont rejetées dans le modèle de référence constituant une source d'information considérable pour les fonctions de la surveillance et pour la conception du modèle de commande.

Modèle de commande

Le modèle de commande représente une utilisation particulière de certains services offerts par l'organe commandé. Il s'agit d'imposer :

- l'ordre d'exécution de certaines activités proposées dans le modèle de référence,
- l'affectation des ressources chargées de les exécuter,
- le contenu des messages échangés entre les modèles de référence et la commande ainsi qu'entre la commande et les niveaux adjacents.

Successivement, au travers des différents niveaux d'abstraction de la structure hiérarchique, une requête de commande de haut niveau est décomposée en une séquence de requêtes plus élémentaires. Un modèle de commande est donc implanté dans chaque bloc concerné par la décomposition de la requête.

Les contraintes déjà exprimées dans le modèle de référence n'ont pas à être réécrites dans le modèle de commande. En revanche, les activités de commande y sont dupliquées. Leur organisation dans le modèle de commande définit la séquence de commande capable de satisfaire la requête émise par le niveau supérieur.

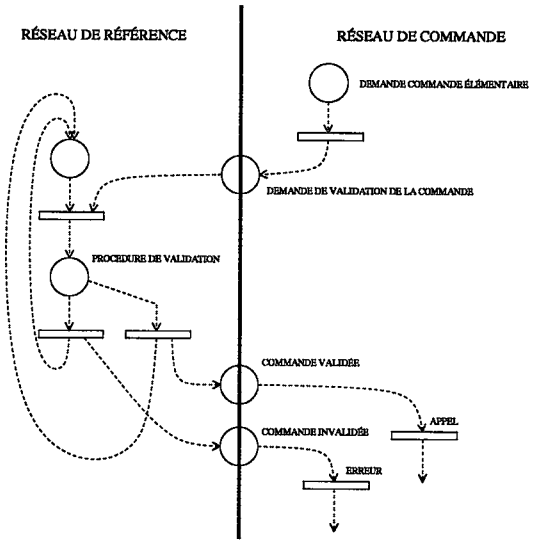

FIG. 3.4 – *Communication Commande - Référence*

Communication Commande - Référence

Puisque les contraintes structurelles sont rejetées dans le modèle de référence, lors de l'exécution d'un appel de commande élémentaire elles doivent être vérifiées. Une activité de commande élémentaire ne peut donc être lancée que si les contraintes imposées par le modèle de référence sont respectées. Ainsi, il existe une transition appelante dans le réseau de commande et une transition appelée dans le modèle de référence. Ce mécanisme est illustré par la figure 3.4.

Par la suite, nous allons nous concentrer sur la spécification du bloc de surveillance.

3.4 Les Fonctions d'un bloc de Surveillance

Les différentes fonctions qui doivent être remplies par un système de surveillance peuvent être définies de la manière suivante. D'abord les situations qui ne correspondent pas à l'exécution de la requête à satisfaire doivent être reconnues : on appelle cette fonction la *détection.* Notons que la fonction détection peut nécessiter l'adjonction de blocs spécialisés fournissant des informations qui n'ont aucune utilité en fonctionnement normal (par exemple la vérification des résultats de l'exécution d'un algorithme). Ensuite, en fonction des informations disponibles, il faut identifier l'origine de cette situation : il s'agit d'une étape de

diagnostic. Puis, en fonction de l'état actuel, il faut élaborer une solution permettant de traiter la défaillance : la fonction de *décision.* Enfin, quand une solution au problème a été élaborée, il s'agit de l'appliquer (ceci comprend l'imposition des évolutions prédéfinies ou le gel de tout ou partie du Micro Système pour des raisons d'urgence ou de sécurité) : c'est le rôle de la *programmation.* Ces quatre étapes permettent de ramener le Micro Système dans un état compatible avec le fonctionnement imposé par les exigences. Enfin, un système de surveillance doit être capable, à tout instant, de maintenir une image aussi fidèle que possible du sous système commandé. Par la suite, nous allons encapsuler chacune de ces fonctions dans des blocs génériques disponibles pour la conception de Micro Systèmes (voir chapitre 2).

3.4.1 Le bloc Détection

La détection des comportements anormaux est une étape délicate. En effet, il faut être capable de classer les situations observables à travers l'interface entre niveaux, comme étant normales ou anormales.

Afin d'être exhaustif, il convient donc de définir l'ensemble des situations anormales comme étant l'ensemble des situations qui ne sont pas reconnues normales. Il suffit alors d'établir les frontières du fonctionnement normal.

La démarche naturelle est de mettre en évidence les différentes classes d'états dans lesquelles le sous système commandé peut se trouver. Etant un système physique, l'ensemble d'états qui peuvent être reconnus à travers de l'interface entre niveaux (états observables) peut se diviser comme suit :

- les *états interdits* à partir desquels des fonctionnements corrects sont en danger d'être violées (saturation d'un capteur, capacité de stockage dépassée, etc.),

- les *états utilisables* sont les états observables non interdits. Tous ces états définissent la zone de fonctionnement normale du Micro Système.

- les *états autorisés* sont les états utilisables appartenant aux séquences normales de commande.

On peut alors, compte tenu de la partition des états d'un système présentée ci-avant, définir deux ensembles de comportements normaux, l'un par rapport au

traitement (flots de matière, d'énergie ou d'information) et l'autre par rapport à la commande en cours d'exécution :

- Vis-à-vis du traitement, tous les états utilisables seront considérés comme normaux,

- Vis-à-vis de la commande, tous les états autorisés par la commande seront considérés comme normaux.

Une situation est définie normale ou anormale par rapport à la commande qui lui est appliquée, et une déviation du comportement spécifié doit toujours être interprétée comme la conséquence d'une défaillance.

Illustrons ceci par un exemple simple. La prise de mesures de température est une situation tout à fait normale par rapport au traitement. Toutefois, si ceci survient alors que le mode commandé est de mesurer l'accélération, le fonctionnement du Micro Système ne peut pas être considéré comme normal.

Une conséquence de cette approche est que, si l'ensemble des états autorisés défini par la commande en cours inclut le traitement d'une défaillance (par exemple, parce qu'elle est assez fréquente), celle ci ne sera pas définie comme une anomalie.

La détection d'une anomalie de fonctionnement se fait dans le réseau de commande (commande élémentaire) par l'analyse des signaux (compte rendus) émis par les blocs fonctionnels. Lorsqu'un compte-rendu active un mécanisme de détection, il constitue un symptôme de dysfonctionnement. Un symptôme est la partie visible d'un comportement qui n'est généralement pas bien connu à l'étape de détection. La phase suivante du traitement de défaillance, le diagnostic, va tester le sous système commandé afin de découvrir la défaillance causant le symptôme. Un exemple de cette situation est illustré par le modèle de vérification et de détection de chaque organe (voir figure 3.5).

3.4.2 Le bloc Diagnostic

Le but du bloc de diagnostic est de rechercher les causes ayant produit une observation particulière. Le diagnostic des situations anormales dans un Micro Système consiste à déterminer quel est le bloc qui, par sa défaillance, a conduit à la détection d'un symptôme de dysfonctionnement (identification du composant défaillant).

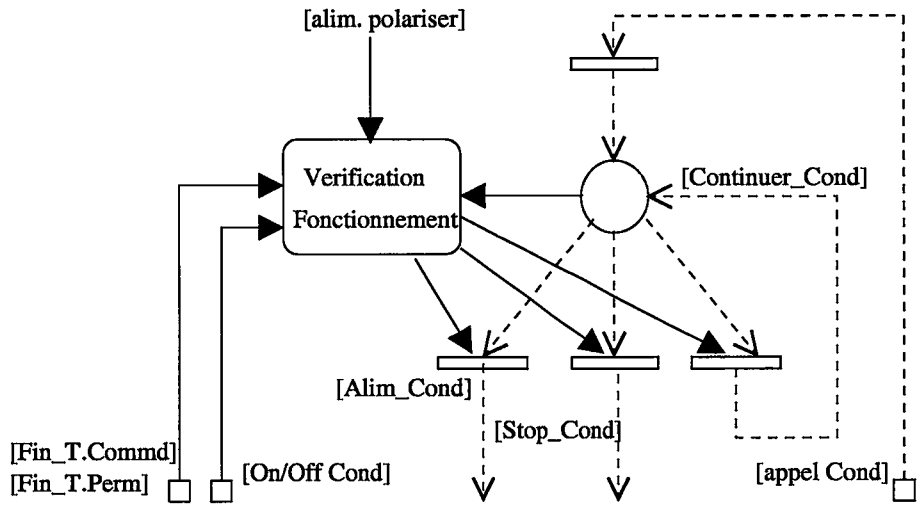

FIG. 3.5 – *Modèle de vérfication et de détection du niveau de commande élémentaire*

Etablir un lien entre le symptôme observé et la défaillance qui l'a provoqué n'est pas évident parce que plusieurs situations peuvent entraîner le même symptôme. Par exemple, une mesure de décélération excessive issue d'un capteur peut être due à un dysfonctionnement (défaillance) ou à une collision du Micro Système avec un élément de l'environnement (pas de défaillance du Micro Système).

La spécification d'un système de diagnostic consiste à énumérer de manière exhaustive les liens de cause à effet entre symptômes et défaillances. Si un lien est manquant, le système de surveillance ne pourra pas trouver la cause de la défaillance. Le modèle de commande surveillance du LAAS propose l'utilisation d'un modèle formel de référence du sous système commandé. La méthode de diagnostic consiste à comparer le fonctionnement effectif du composant suspecté avec le comportement observé sur le modèle de référence. Si une différence est observée, le composant suspecté est hors fonctionnement. Par exemple, la vérification d'un bloc capteur est faite en faisant un test du capteur et en comparant sa réponse avec la mesure suspecte. Cette méthode permet un diagnostic structuré du sous système commandé. Ainsi, un bloc encapsule d'autres blocs plus élémentaires dont les modèles ne sont utilisés que si le bloc global est déclaré hors fonctionnement. La mise en oeuvre nécessite l'obtention d'un modèle formel du sous système commandé exploitable par le système de diagnostic et

la méthode peut devenir impraticable si de nombreuses vérifications sont nécessaires pour tester un bloc ou composant. Il semble intéressant à ce stade de définir un arbre hiérarchisé de fautes [65]. Cette méthode aide au concepteur à gérer la complexité du diagnostic et en même temps donne les critères pour la modélisation des fautes pendant la conception aval.

L'activation du bloc diagnostic peut être effectuée de trois façons :

- sur demande : l'unité de contrôle veut avoir un diagnostic sur l'état général du système,
- cycliquement : des tests sont effectués sur l'ensemble des parties sensibles pour assurer une fiabilité permanente,
- ponctuellement : à la suite de la détection d'une mesure en dehors d'un domaine de fonctionnement nominal.

Le rapport de diagnostic comprendra des résultats de tests des organes ainsi que le mode de fonctionnement courant. Si le diagnostic ne dévoile pas une situation anormale, le Micro Système revient dans un mode de fonctionnement précédent l'activation du bloc diagnostic. Sinon, si des anomalies sont détectées, des messages d'alarme avec le rapport de diagnostic pourront être envoyés à l'unité de contrôle et le Superviseur pourra prendre l'initiative de passer à d'autres modes de fonctionnement selon les cas:

- si tous les organes sont détériorés ou si le niveau d'alimentation est trop bas,
- si le niveau d'énergie est convenable et si au moins un organe fonctionne correctement.

3.4.3 Le bloc Décision

Ce bloc prend des décisions qui permettent de résoudre les conséquences d'une défaillance d'un composant du sous système commandé.

Si l'origine de la défaillance a été identifiée (le bloc défaillant a été identifié), deux décisions doivent être prises :

- L'action sur le sous système commandé : il s'agit d'isoler le bloc défaillant pour garantir une opération quand même.

- L'action sur le bloc commander : il s'agit de synchroniser le modèle du bloc de commande sur l'état réel du sous système commandé. Il faut ensuite décider de l'état vers lequel le sous système commandé doit évoluer pour retrouver un mode de fonctionnement normal en utilisant au mieux les fonctionnalités restantes du sous système commandé.

Si la défaillance n'a pas été identifiée, le bloc décision doit propager le traitement de défaillance en retournant un compte rendu vers le niveau supérieur indiquant par une alarme le symptôme détecté et l'impossibilité de trouver un composant défaillant. Le niveau supérieur doit, par la suite, replacer le bloc propageant dans un état connu.

3.4.4 Le bloc Programmation

Le bloc de Programmation doit se charger de reprogrammer le mode de fonctionnement du sous système commandé en accord avec les actions issues du bloc de décision :

- modifier ou remplacer le modèle de référence du bloc de commande,
- reprendre le fonctionnement normal en remplaçant le bloc défaillant si des blocs redondants ont été prévus (mode intermédiaire),
- reprendre le fonctionnement normal en isolant le bloc défaillant si les ressource restantes le permettent (mode intermédiaire),
- arrêter le fonctionnement normal si aucune ressource ne reste disponible pour un fonctionnement normal (mode dégradé),
- exécuter des action d'urgence en imposant un comportement prédéfini au sous système commandé.

A tout moment, le niveau de gestion peut interrompre les activités en cours du Micro Système (interruption de plus haut niveau). Dès que le niveau surveillance - coordination reçoit une commande, il interrompt ses activités en cours pour exécuter la commande prioritaire. Ce bloc peut aussi être activé à la suite d'un diagnostic qui a mis en évidence un dysfonctionnement sur les organes. Le bloc de Programmation permet de changer de mode de fonctionnement courant (résultant d'une nouvelle commande ou d'un diagnostic), d'exécuter des séquences de tests et renvoyer les résultats correspondants, de transmettre des

historiques (mesures, anomalies, etc.), de reconfigurer les différents paramètres des organes et enfin de reprogrammer l'horloge interne (temps universel).

3.4.5 Le bloc Suivi

Le bloc suivi doit recueillir l'ensemble des informations émanant d'un organe, afin de maintenir une image aussi fidèle que possible du système commandé au sein du système de surveillance. Le bloc suivi reçoit tous les comptes rendus issus du niveau de commande élémentaire et maintien l'état courant de l'organe sous surveillance.

3.4.6 Fonctionnement de la surveillance

Le traitement de base d'une défaillance consiste à enchaîner successivement les blocs de surveillance telles que la détection, le diagnostic et la programmation. La figure 3.6 nous donne la représentation de la stratégie adoptée.

La stratégie et le fonctionnement du superviseur integré sont les suivants :

- Tout d'abord le Micro Système est initialisé. L'initialisation implique l'alimentation de toutes les organes, l'envoi de jetons aux places correspondantes à chaque organe pour initialiser son exécution (jeton "APPEL");

- Le système initialisé, chaque organe réalise une auto-vérification de son fonctionnement. Si le résultat de la vérification est correct, alors un report est envoyé au superviseur et l'organe continu actif (jeton "END-REPPORT"). Si un disfonctionement est detecté, l'organe entre un état d'attente (jeton "WAIT") et le diagnostic commande les tests nécessaires pour connaître l'état de tous les composants des organes (jeton "BEGIN-TESTS").

- Après achèvement des tests, un rapport est envoyé au diagnostic pour son analyse (jeton "TEST-REPPORT");

- En recevant l'analyse du diagnostic, le bloc décision cherche la stratégie à suivre : le mode de fonctionnement le plus adapté, les organes à programmer et les données de configuration à envoyer aux organes choisiés (jetons "KEEP-RUNNING" ou "STOP"). Le bloc décision déclenchera la programmation de la stratégie choisie. Les modes de fonctionnement sont mutuellement excluants ;

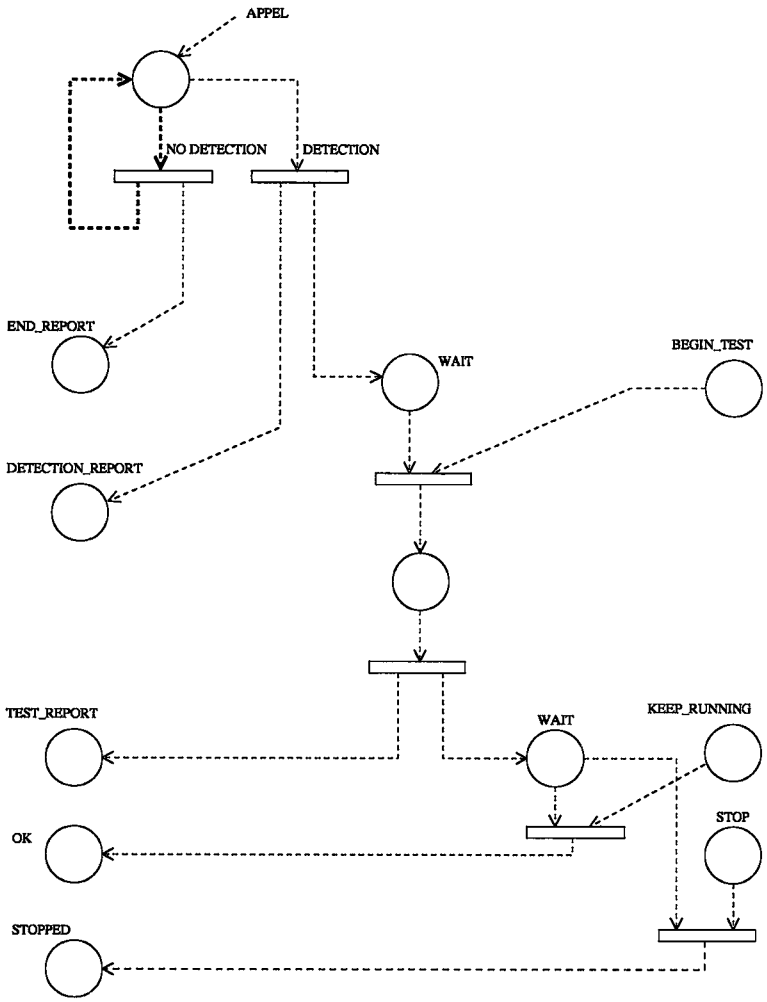

FIG. 3.6 – *Stratégie de Surveillance*

- La programmation de chaque mode de fonctionnement demande avant tout la coupure de l'alimentation aux organes n'appartenant pas à la stratégie choisie. Ensuite, la programmation envoie l'information d'initialisation aux blocs et aux temporisateurs correspondants. En recevant l'accusé de réception des données d'initialisation, le superviseur va à une place de surveillance. Il attend la détection d'un des deux événements: le symptôme de défaillance ou la fin du temps d'opération du mode d'opération courant;

- À la détection d'un événement sur un organe, le superviseur force le signal de temporisation des autres organes en leur indiquant que l'exécution est terminée et qu'ils doivent s'arrêter. Le signal de détection implique que le jeton et l'information du symtôme detecté soient retournés au bloc diagnostic. En même temps le superviseur retire tous les jetons des organes actifs en envoyant le signal de fin de temporisation;

- Le diagnostic analyse l'information de détection et il commande les tests nécessaires pour envoyer un rapport au bloc de décision. Ainsi, le bloc de programmation peut éxecuter à nouveau la procédure d'initialisation et de configuration du mode de fonctionnement choisie par la décision;

Le fonctionnement général de chaque organe est le suivant: le superviseur est chargé de faire la demande de l'organe correspondant pour que celui ci démarre son activité. Ceci est materialisé par l'envoi du jeton d'appel et du signal d'alimentation. L'organe vérifie qu'il n'existe pas une demande d'arrêt venue du superviseur (signal de fin de temporisation). En cas de demande d'arrêt, l'organe envoie le jeton au superviseur indiquant l'arrivée de l'événement fin d'éxecution par fin de temporisation. S'il n'existe pas une demande d'arrêt, l'organe déclenche l'éxecution de ses activités et ensuite, il vérifie en interne le fonctionnement correct de ses blocs. Tant qu'aucune détection n'en est issue, l'organe continuera son fonctionnement en permanence.

Pour interdire la possibilité que le superviseur envoie un signal d'initialisation à un organe déjà actif, nous avons disposé une place marquée dans le réseau de coordination de l'organe, laquelle permettra l'initialisation de celui ci mais empêchera que le superviseur ré-initialise un organe déjà en exécution.

À la détection d'un événement, le jeton indicant "organe actif " retourne à sa place pour montrer que l'organe est prêt à être initialisé à nouveau. Ce jeton

sera placé la chaque fois que le Micro Système est initialisé et en constitue une des activités de coordination du superviseur.

Il est possible qu'il existe plusieurs types d'appel à un organe et plusieurs événements à détecter. Ceci dépend du fonctionnement particulier de l'organe spécifique.

La coordination d'un organe de la part du superviseur peut être simple ou complexe selon la fonctionalité demandée, mais le principe reste le même.

3.4.7 Les Flots d'information dans l'organe Superviseur Intégré

Une réactivité du micro système est présente à tout instant avec l'environnement. Un micro système de surveillance doit prendre des mesures des quantités physiques, en s'adaptant aux conditions externes (auto-configuration) : l'organe superviseur integré configure le micro système activant les organes correspondants (capter, conditionner, traiter, etc.). Les mesures sont mémorisées seulement si la variation par rapport à la valeur précédente dépasse une valeur relative afin d'éviter une saturation prématurée de l'organe de mémorisation par l'encombrement de mesures inutiles et de ne conserver que celles qui sont pertinentes. Par la suite, si cette donnée est mémorisable, on détermine si la mesure appartient à un domaine normal de fonctionnement ou est en dehors de celui-ci. Si la mesure est hors du domaine normal, une détection est déclenchée pour demander d'établir un diagnostic sur l'état du système afin de vérifier que la mesure retranscrit correctement un événement particulier et n'est pas due à une défaillance d'un des capteurs. Dans ce cas, le superviseur génère une alarme, à laquelle une vignette de temps universel est adjointe et ensuite elle est mémorisée. Sinon, si la donnée indique un fonctionnement normal, une vignette temporelle lui est adjointe aussi permettant d'indiquer sa date d'acquisition. A partir de là, elle est envoyée vers un bloc de mémorisation qui conservera ces mesures afin d'établir des historiques de mesures.

Concernant les blocs de mémorisation des mesures et des alarmes, seule une condition permet de les vider vers l'unité de contrôle : sur demande de l'unité de contrôle éloignée.

Cependant, avant d'être transmises, les données passent par un bloc protocole d'émission qui associe à ces informations des champs de bits particuliers pour leurs mises en forme (code correcteur d'erreurs, code d 'identification, etc.).

Aux modes commandés par l'unité de contrôle correspond une commande particulière, celle-ci est envoyée par le bloc superviseur et va permettre de configurer les différents organes de telle sorte que le mode de fonctionnement désiré soit atteint.

L'unité de contrôle gère le fonctionnement global du Micro Système même si celui-ci est autonome. Par conséquent, l'unité de contrôle peut à tout instant interrompre les activités en cours en lui demandant d'effectuer des opérations particulières. Ces opérations peuvent être :

- séquences de tests avec le renvoi des résultats correspondants,
- changement de mode de fonctionnement,
- renvoi d'historiques (mesures et/ou alarmes),
- configuration des paramètres des organes (seuils, horloges, etc.),
- réinitialisation.

Ainsi, le Micro Système reçoit des commandes qui sont traitées par des blocs spécialisés avant d'arriver au superviseur pour leur interprétation. À partir de là, le superviseur envoie les séquences de commandes qui doivent être exécutées par le Micro Système.

Ainsi, trois flots ont été identifiés :

- le flot descendant (requêtes issues du niveau de gestion),
- le flot ascendant (alarmes, mesures),
- le flot interne.

Dans le premier cas, les informations issues du niveau gestion (requêtes) seront constituées des données suivantes :

- origine : niveau gestion ;
- service demandé par le niveau de gestion. Il peut donc s'agir d'une requête de commande, d'une requête de test, d'une requête de programmation ou d'une requête de configuration ;
- données fonctions : elles sont à définir par rapport aux besoins de ces fonctions.

Dans le deuxième cas, les informations issues du niveau inférieur, véhiculeront les données suivantes :

- origine : niveau surveillance coordination ;
- nom du bloc : dans ce cas, la donnée nom du bloc est suffisante pour permettre au superviseur de sélectionner le bloc concerné par l'information ;
- données fonctions : elles sont à définir par rapport aux besoins de ces fonctions.

Dans le troisième cas, les informations issues des blocs de surveillance - commande véhiculeront les données suivantes :

- origine : interne
- nature :
  - requête : cette donnée correspond à une requête vers le niveau de commande élémentaire,
  - compte rendu : cette donnée correspond à la fin d'exécution d'un service demandé. Par exemple, si un diagnostic a été lancé dans un bloc à partir du niveau gestion, la fin d'exécution de ce diagnostic correspondra à l'envoi vers le bloc de programmation d'un compte rendu d'exécution,
  - résultat : cette donnée correspond au résultat fourni par le bloc (service).

3.5 Fonctions d'Autonomie de l'organe Superviseur Intégré

L'organe superviseur intégré doit assurer par ses fonctions, l'autonomie du Micro Système. L'autonomie est vue comme la capacité de surveiller, d'administrer ses propres modes de fonctionnement, de s'adapter à l'évolution de son environnement et de communiquer avec le niveau de gestion. Ainsi, le Micro Système prend la configuration qui lui donne le qualificatif de bloc *organisme.*

Modes de Fonctionnement

En vu d'une autonomie embarquée, nous avons défini le bloc programmation (des modes de fonctionnement), trois modes de fonctionnement selon l'état interne du Micro Système et deux modes de fonctionnement selon l'état de l'environnement externe du Micro Système. Ainsi du point de vue de l'état interne, si tous les organes du Micro Système sont opérationnels, on parle de mode *Normal* ; s'il reste au moins une fonction assurée , alors on parle de mode *intermédiare* ; et finalement si aucune fonction ne peut être assuré ou le niveau de l'alimentation est trop bas, le mode sera alors appelé *dégradé.* D'autre part, du point de vue de l'état de l'environnement du Micro Système, si le système surveillé est au repos, on parle du mode *veille* ; si le système sous surveillance est actif, alors on parle du mode *permanent.*

La gestion des modes de fonctionnement est autonome et lors d'une communication avec le niveau de gestion, le Micro Système revient dans le mode précédent l'interruption dès la fin du renvoi des informations (résultats tests, historiques, etc.) vers l'unité de contrôle éloignée.

Mode Normal Ce mode normal comporte deux sous modes qui caractérisent un fonctionnement normal du système. Ici, toutes les parties sensibles telles que les capteurs, les actionneurs, l'électronique ou l'alimentation sont en parfait état de marche.

Fonctionnement Permanent Normal : Le superviseur intégré envoie une commande autorisant l'alimentation de tous les organes nécessaires de type capter, conditionner, traiter, memoriser, dater, etc. Ce mode est caractérisé par le prélèvement, l'analyse et la mémorisation en permanence de mesures sur l'environnement observé.

Fonctionnement Veille Normal : C'est le mode de fonctionnement par défaut, c'est à dire que le Micro Système se trouve dans ce mode après une réinitialisation du dispositif. Dans ce type de fonctionnement, seul le niveau surveillance - coordination est alimenté ; tous les autres sont au repos. Lorsque le système surveillé est inactif, la prise de mesures n'est pas nécessaire, le Micro Système se place en mode veille (le bloc de programmation envoie la commande autorisant ce passage). Sachant que le bloc communiquer reste alimenté en permanence, il attend l'arrivée de commandes éventuelles de l'unité de contrôle

éloignée.

Mode Intermédiaire Ce mode est équivalent au mode normal. Cependant, ici, le niveau d'énergie est aussi convenable mais il y a au minimum un organe qui fonctionne correctement et un organe détérioré. Ce mode est atteint après avoir détecté un dysfonctionnement sur un ou plusieurs organes. Avant de poursuivre les mesures, il faut localiser les données correspondant aux organes défectueux pour ne plus les sélectionner. Dans ce mode intermédiaire, il y a aussi deux sous-modes qui ont déjà été décrits précédemment:

- mode permanent intermédiaire
- mode veille intermédiaire

Ainsi, la différence entre ces sous-modes et ceux appartenant au mode normal est le nombre d'organes détériorés.

Mode Dégradé Le Micro Système se trouve dans ce mode dégradé à la suite de tests qui indiquent que le niveau d'énergie disponible est insuffisant pour assurer un fonctionnement nominal du système et/ou une défaillance des organes (détériorés ou fournissent des mesures non représentatives de la réalité). Si seul le niveau d'énergie est en cause, il est possible de recharger le Micro Système et de le réinitialiser.

Mode Réinitialisation

Comme son nom l'indique, ce mode permet de réinitialiser le système lors d'une première utilisation, ou à la sortie d'un mode dégradé si seulement le niveau d'alimentation était trop bas et qu'au moins un organe fonctionne correctement, ou encore sur demande de l'unité de contrôle. La procédure de réinitialisation est la suivante:

- les différentes mémoires vives (utilisées pour des calculs intermédiaires) des organes sont remises à zéro,
- le Superviseur récupère dans sa mémoire les différentes valeurs de configuration pour tous les organes,
- le bloc programmation envoie une commande permettant le passage en mode veille normal afin de patienter jusqu'à la réception d'une nouvelle

commande provenant de l'unité de contrôle ou que le système surveillé s'active.

Des que le mode veille est atteint, le dispositif est à nouveau opérationnel.

3.6 Conclusions

Les techniques actuelles de testabilité utilisent souvent le standard IEEE 1149.1-1990, développé par le "Join test Action Group" (JTAG) en 1980. Cette technique est basée sur le test de frontière ("Boundary Scan Test" - BST) permettant aux ingénieurs de faire un test des dispositifs déjà montés sur un circuit et de les programmer. La procédure de test est la suivante:

- introduire le patron de test à l'intérieur des registres de frontière (BST),
- faire passer ces signaux au travers du dispositif sous test,
- acquérir les signaux de réponse au travers des registres de frontière (BST),
- déplacer les signaux sur les registres de frontière pour les comparer par rapport au patron d'entrée.

Dans le contexte de la conception amont des applications envisagées et de leurs exigences de fiabilité, de sûreté de fonctionnement et d'autonomie, l'adjonction d'un système de test plus évolué est indispensable.

Dans ce chapitre, nous avons proposé un *système de surveillance en temps réel de toute l'électronique embarquée* basé sur une architecture de commande hiérarchisée à trois niveaux: le niveau de commande élémentaire, le niveau de surveillance - coordination et le niveau de gestion. Le but d'un tel système étant la recherche des composants en défaillance, la gestion de l'autonomie du Micro Système et la communication par des alarmes des dysfonctionnements. La réussite de cette activité dépend fortement du degré de testabilité du système électronique, et nous avons proposé une structure qui permet de:

- valider la commande par modèle de référence,
- vérifier le résultat d'une commande après son exécution,
- et enfin, tester les organes responsables de l'exécution d'une commande.

Nous avons défini les modes de fonctionnement d'un Micro Système autonome et les différents conditions et contraintes de son administration.

En outre, les flots d'information entre niveaux ont été caractérisés et définis.

Cette architecture générique donne lieu à un superviseur intégré (niveau surveillance - coordination) qui pourra être personnalisé à l'application, par l'utilisation des blocs génériques que nous avons proposés (détecter, diagnostiquer, tester, décider et programmer). Ces blocs pourront être imbriqués à la commande primaire pour surveiller son état et sa sûreté de fonctionnement.

Chapitre 4

CONCEPTION D'UN DISPOSITIF MULTI CAPTEURS INTEGRE D'ENREGISTREMENT DES CONTRAINTES ENVIRONNEMENTALES

4.1 Position du problème

La méthodologie structurée de conception de Micro Systèmes et l'architecture de commande - surveillance proposées vont être illustrées au travers d'un cas d'étude.

L'homme a crée des systèmes et dispositifs complexes par leur structure et leur comportement ainsi que par la façon de les contrôler, les administrer, les entretenir, les faire évoluer et s'adapter aux changements environnementaux et technologiques. La surveillance de ces systèmes s'avère nécessaire dans la mesure où la prévision exhaustive de tous leurs comportements dans tous les cas possibles d'utilisation et de leur état de vieillissement est difficile.

Par exemple, la structure mécanique d'un avion devient de plus en plus flexible et devient une structure molle. Et ceci avec l'intervention de plusieurs fabricants (fuselage, turbines, avionique, etc.). Le contrôle de cette structure n'est pas possible sans la connaissance de sa dynamique pendant le vol. Ceci demande l'observation en temps réel de l'accélération et de la température de chacun de ses éléments. D'autre part, la surveillance de son état de vieillissement dans le temps ou après un fort effort pourrait mieux orienter les activités de maintenance au sol et même éviter des accidents catastrophiques dus à la méconnaissance de l'état global de l'avion.

4.1.1 Définition du problème

Dans le cas d'étude choisie, le problème que l'on se pose est comment surveiller une structure mécanique complexe. A partir d'une première analyse, le problème le plus difficile n'est pas l'acquisition des mesures d'accélération et de température mais:

- le nombre élevé de capteurs qu'il faudrait mettre en place pour bien établir la dynamique de l'avion;
- les restrictions de poids de l'avion;
- la fiabilité de la solution face aux contraintes de type environnemental (basse température, interférence électromagnétique, etc);
- la solution retenue doit être à maintenance curative (non réparable);
- l'autonomie vis a vis de l'alimentation électrique;
- la sûreté de fonctionnement malgré des longues périodes de veille et de fonctionnement;
- la sensibilité et l'exactitude des capteurs;
- la miniaturisation poussée au delà de 50x30x20 mm;

Les objectifs que l'on espère atteindre avec la solution recherchée sont:

- Définir un concept de famille de produit et les outils nécessaires à son développement,

- Réaliser des prototypes et les intégrer dans des applications relevant des domaines de la surveillance de l'industrie aéronautique,
- Établir les niveaux de qualité, intégrité et fiabilité de ces prototypes,
- Développer les éléments nécessaires à une miniaturisation poussée de ces dispositifs.

4.1.2 Spécification des besoins

Les Dispositifs d'Enregistrement des Contraintes (D.E.C) sont des circuits multicapteurs capables de mesurer des paramètres caractérisant l'usage de différentes parties d'un système mécanique opérationnel. Cette surveillance permettra de déterminer avec plus de précision les opérations de maintenance préventives à effectuer, et de servir de système d'alarme afin d'assurer la sûreté de fonctionnement du circuit observé. Ils sont particulièrement utiles dans les systèmes de transport, mais d'autres usages dans le domaine du génie de la santé sont envisagés. Concernant l'application aéronautique, l'idée consiste à mesurer les conditions d'environnement et à en répertorier les valeurs les plus significatives qui entraînent des accélérations du processus de vieillissement : températures très élevées, contraintes mécaniques extrêmes, vibrations répétées, etc. Sur la base de ces enregistrements, il sera alors possible de choisir d'accélérer les remplacements de pièces ou à l'inverse de temporiser si les contraintes ont été très supportables. Le concept, né il y a quelques années pour la maintenance des équipements militaires, avait donné lieu à des réalisations unitaires, lourdes et onéreuses. Ce concept revient maintenant sous la forme d'objets miniatures, standards et produits en série.

4.1.3 État du domaine

Actuellement, les dispositifs d'enregistrement des contraintes sont disponibles sous forme d'équipements de taille moyenne (100 x 70 x 40 mm). Les évolutions prévisibles devraient mener à des dimensions de l'ordre de 50 x 30 x 20 mm sans qu'il y ait besoin de franchir de barrière technologique. Des travaux antérieurs ont déjà permis d'apprendre à utiliser ces produits, d'en connaître les limites et la tenue à l'environnement. Des intégrations sur des systèmes

complexes ont aussi été réalisées. Toutefois, l'intégration à des équipements aéronautiques ou l'utilisation pour le suivi des colis de valeur nécessitent une miniaturisation plus poussée. Le volume visé devrait être équivalent à celui d'un composant électronique pour pouvoir être facilement reporté sur une carte à puce ou une carte d'équipement électronique. Quant à l'offre industrielle, peu d'informations sont diffusées concernant l'avancement de ce type de projets. Aux Etats-Unis tout d'abord, des sociétés développent de tels systèmes pour des applications spécifiques aux courses automobiles (équivalent de la formule 1) mais ces équipements ont encore un volume, un poids et une consommation très élevés. En Europe, Siemens Automotive développe des dispositifs moins coûteux pour des applications dans l'industrie automobile. En France, quelques sociétés effectuent des recherches sur ce thème; ce sont principalement les sociétés Sensorex, Befic et Acorvitas. En comparant les systèmes disponibles sur le marché, les principaux points négatifs qu'il en ressort concernent *l'encombrement des circuits* (encore très volumineux) et la *consommation énergétique élevée*. L'étude menée ici va permettre de palier ces problèmes en proposant un *Micro Système* basé sur une *électronique faible consommation*, remplissant des fonctions d'observation, capable de prendre des décisions et communiquant avec l'extérieur, tout en assurant une fiabilité de fonctionnement permanente.

4.1.4 Conceptualisation des solutions possibles et exploration de l'espace de conception.

La conceptualisation des solutions doit définir tout d'abord l'espace dans lequel résident toutes les solutions possibles. Cet espace doit ensuite être limité par le niveau de connaissances et par les contraintes imposées par l'application. Les contraintes fictives doivent être dépistées et éliminées.

L'ensemble de connaissances est très large:

- La conception de Micro Systèmes,
- La conception et la fabrication de structures miniaturisées en 3D à base de composants électroniques,
- La conception et la fabrication de systèmes électroniques (systèmes mixtes analogues et digitaux, capteurs et communications),
- Les équipements aéronautiques,

- L'analyse, la validation et la vérification en environnement sévère.

Pour mieux rechercher une structure ou architecture adaptée aux besoins, l'espace des solutions possibles doit être limité davantage. L'espace étant multidimensionnel, chaque dimension doit correspondre à une caractéristique du système à concevoir, à un critère d'évaluation ou à une structure fondamentale. L'espace de conception peut donc comprendre des sous espaces fonctionnels et structurels. La méthodologie de conception définit des règles permettant de passer d'un sous espace à l'autre pendant l'évolution du processus de conception, tout en respectant les compromis entre eux. Ces dimensions doivent être générées à partir d'une analyse soignée et très détaillée afin de ne prendre en compte que les véritables contraintes [37] [43].

Les dimensions de l'espace de conception des D.E.Cs peuvent être définies à partir d'une liste de caractéristiques importantes:

- La taille,
- La robustesse,
- La fiabilité,
- L'autonomie,
- La consommation de puissance,
- La communication sans fil,
- L'adaptabilité aux nouvelles technologies ou interopérabilité,
- L'interchangeabilité de blocs ou modularité,
- L'adaptabilité aux besoins,
- L'intelligence locale.

Nous pouvons constater que l'autonomie du D.E.C augmente si la consommation de puissance est basse et le niveau d'intelligence locale est élevé. Plus le D.E.C est autonome, moins il a besoin de communiquer. Un niveau élevé d'intelligence locale permettra au D.E.C de mieux s'adapter aux nouvelles technologies, augmentera son niveau de modularité et d'adaptabilité aux besoins, mais accroîtra aussi la consommation de puissance électrique. Les critères les plus importants

pour cette application sont sans doute la fiabilité et la robustesse qui sont des concepts très proches et corrélés. Par contre la taille des D.E.C (3 cc) n'est plus une contrainte par rapport au poids du système surveillé. Après cette analyse, nous pouvons conclure que les dimensions de l'espace de conception des D.E.Cs sont:

- la fiabilité, robustesse et sûreté de fonctionnement;
- l'autonomie vue comme le niveau d'intelligence locale, la capacité de communiquer et la modularité;
- la consommation d'énergie.

Ces dimensions correspondent aux critères d'évaluation des architectures proposées. Leur analyse permettra d'identifier le degré d'importance relative et pour le montrer un poids normalisé devra être assigné à chaque critère.

4.2 Génération d'une description architecturale

4.2.1 Partition par flots d'Energie, Matière et Information

Nous pouvons constater la présence des quatre domaines: le flux d'air pour pouvoir mesurer la température extérieure, les flux d'énergie mécanique (accélérometrie), le flux d'énergie électromagnétique (RF / optique) et le flux d'information. Cette identification nous permet de chercher dans la librairie des options différentes pour modéliser l'architecture du D.E.C.

Le D.E.C comporte trois types de capteurs:

- Capteurs accéléromètres trois axes qui permettent, entre autre, de mesurer ou de contrôler le comportement au niveau des vibrations de certains systèmes aéronautiques ou les efforts de torsion pouvant endommager des parties mécaniques de l'avion (fuselage, aile,..),
- Capteurs de température pour étudier et évaluer les cycles extrêmes auxquels ont été soumis les différents systèmes observés,
- Capteurs électromagnétiques pour caractériser le champ magnétique environnant (décharges électrostatiques, CEM, etc.).

Pour accomplir sa fonction, ce dispositif aura:

- accès à différentes valeurs dans le système environnant,

- une capacité propre de calcul,

- la possibilité de générer des tests, des commandes et de conserver en mémoire des informations pertinentes,

- une capacité à communiquer avec l'extérieur.

Le plus grand nombre possible de D.E.C devront être contrôlables à partir d'une seule unité de contrôle. Le transfert de données sans fil entre un D.E.C et son unité de contrôle devra être assuré même en présence d'un obstacle sur le parcours de transmission (en utilisant par exemple une réflexion diffuse du faisceau de la source dans le cas d'une communication optique). Ces D.E.C ne pourront communiquer qu'avec l'unité de contrôle. La consommation d'énergie étant un critère primordial, les différents systèmes en présence communiqueront de façon asynchrone, c'est à dire, un système ne sera alimenté qu'à partir du moment où il sera sollicité. Pour assurer un bon fonctionnement en permanence, le Superviseur Intégré pourra effectuer des tests sur les capteurs ainsi que sur l'alimentation. Chaque D.E.C devra fournir au moins les informations suivantes :

- trois valeurs d'accélération (selon les trois axes de mesures),

- une lecture de la température (au voisinage des accéléromètres),

- une mesure concernant l'environnement électromagnétique de proximité,

- un code d'identification.

Les mesures auront une résolution de 8 bits (précision de l'ordre de quelques pourcentages sur la mesure). La mesure d'un paramètre est faite au plus toutes les 100 ms, soit une fréquence de 10Hz (périodicité correspondant au temps de réaction d'un être humain qui ne peut pas effectuer d'opérations de correction en cas d'anomalies dans un délai inférieur à 100ms). Cinq paramètres physiques doivent être étudiés, d'où la fréquence de travail de 50 Hz (5x10 Hz). D'après le théorème de Shannon, la fréquence d'échantillonnage doit être supérieure ou égale à deux fois la fréquence du signal, d'où sa valeur : 100 Hz (2x50 Hz). Les dimensions de ce D.E.C sont les suivantes : 1cm x 3cm x 5mm.

4.2.2 Spécification du fonctionnement et de l'architecture du Système de Surveillance Répartie

Chaque D.E.C sera disposé dans des endroits stratégiques soumis à des contraintes d'environnement extrêmes et nécessitant une surveillance particulière (par exemple : les ailes ou le fuselage d'un avion).

Fiable, il pourra fonctionner selon plusieurs modes : *Normal (permanent, commandé, veille), Intermédiaire, Programmation, Diagnostic, Dégradé et Réinitialisation et pourra communiquer avec une unité de contrôle éloignée.*

En fonctionnement normal, des mesures seront prélevées sur l'environnement mécanique observé. Après traitement, le D.E.C. pourra établir un diagnostic, envoyer des messages d'alarme à l'unité de contrôle s'il juge que certaines mesures sont critiques, mémoriser celles qui sont pertinentes et établir des séquences de commandes à exécuter (autotests, changement de modes, etc.). Ainsi, lorsque les services de maintenance des équipements effectueront les contrôles de routines des appareils, ils pourront très rapidement avoir accès aux informations pertinentes relatives aux conditions de vols et sauront si une anomalie a été détectée pendant le dernier vol.

De la même manière, lors d'opérations de maintenance plus approfondies, ils pourront récupérer les informations de tous les D.E.C sur un terminal portable et tracer l'historique des contraintes subies par les systèmes mécaniques surveillés. Ainsi, à partir de l'étude de ces résultats, la décision pourra être prise d'anticiper le remplacement de certains équipements ayant été soumis à de trop fortes contraintes environnementales, ou alors de prolonger l'utilisation si ces contraintes ont été très supportables.

En outre, lors de ces mêmes opérations, les équipes de maintenance pourront effectuer des tests sur tous les D.E.C afin d'assurer une fiabilité de fonctionnement permanente ou de détecter et localiser une panne dans le système. Pour cela, deux parties distinctes vont être testées : les micro capteurs et la partie électronique. Pour les micro capteurs, une commande spécifique est envoyée par le technicien par l'intermédiaire de l'unité de contrôle vers tous les D.E.C leur indiquant de passer en mode diagnostic. Ensuite, des actionneurs exciteront les différents capteurs, leurs réponses seront comparées à des valeurs de référence et les résultats de ces comparaisons indiqueront l'état des circuits. Concernant la partie électronique, il suffira de transmettre une requête à tous les D.E.C afin qu'ils renvoient la date et l'heure courante qu'ils ont en interne. Cette procé-

dure permettra de tester tous les éléments composant la partie électronique : la chaîne de réception, le programateur des modes de fonctionnement (superviseur integré), le bloc dater des données prélevées (comportant l' "Universal Time Coordinated" de référence) et enfin la chaîne d'émission. Ainsi, en comparant l'heure transmise par le D.E.C et l'heure précise courante, il sera possible de détecter la présence d'anomalies.

Le nombre sans cesse croissant d'applications dans les divers secteurs de l'industrie nous oblige à développer un système totalement autonome d'un point de vue énergétique et de fonctionnement *(Structure Déportée)*. Autrement dit, les dispositifs à étudier doivent intégrer leur source d'alimentation, avoir une capacité propre de calcul et pouvoir communiquer avec l'extérieur sans fil. De la même manière, la grande variété des applications (spatiales, aéronautiques, domotiques, industrielles, médicales,...) implique que le nombre de paramètres physiques à surveiller ne soit jamais constant, c'est à dire que le nombre de capteurs pourra varier d'une application à l'autre. Pour cette raison, il faudra développer une *architecture* qui pourra facilement être *modifiée selon* le contexte *d'utilisation*, les modifications s'effectuant principalement sur la partie logicielle du circuit.

Les différentes parties d'un système de surveillance comportant un certain nombre de capteurs (fusion multisensorielle) font appel à des technologies et des savoir-faire d'origines pluridisciplinaires qui ont pour conséquence l'augmentation de la complexité du processus de conception. Pour les systèmes dédiés à la surveillance d'équipements mécaniques, nous pouvons distinguer quatre parties essentielles (voir figure 4.1) :

- un système mécanique à surveiller,
- l'environnement où ce système évolue,
- un nombre de D.E.C. localisés à des endroits stratégiques du système mécanique,
- une unité de contrôle éloignée.

4.3 Conception de l'Architecture d'un D.E.C.

Nous pouvons décrire les interactions (figure 4.2) entre les D.E.C., leur environnement et une unité de contrôle éloignée de la manière suivante :

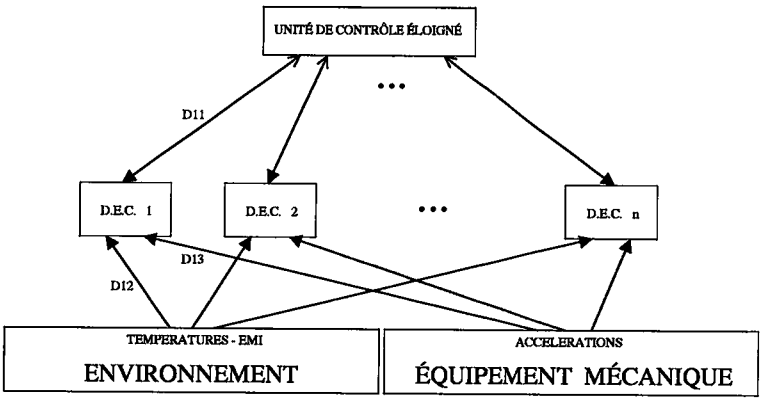

FIG. 4.1 – *Esquisse Architecturale du Système de Surveillance*

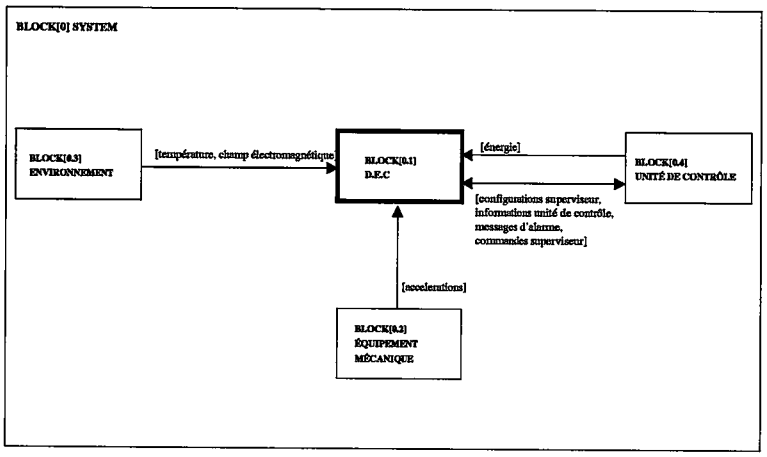

FIG. 4.2 – *Interface du Dispositif d'Enregistrement des Contraintes*

Désignation	Type	Description
Energie (RF/Optique)	Entrée	L'énergie reçue par liaison RF ou Optique permettra de recharger une micro pile dont la fonction est d'alimenter le dispositif. Cette énergie sera distribuée à tout le système sous le contrôle d'un Superviseur. La présence de plusieurs convertisseurs commandés par des contacts (micro relais) permettra d'alimenter les circuits dont les tensions sont différentes.
Commandes Superviseur	Entrée	Le D.E.C est un dispositif autonome, autrement dit, il pourra gérer lui même son fonctionnement. Cependant, l'Unité de Contrôle peut à tout moment interrompre les activités du D.E.C en lui envoyant des commandes pour changer le mode de fonctionnement courant, générer des tests et renvoyer les résultats correspondants afin de vérifier le bon déroulement des opérations, obtenir des informations concernant l'environnement observé (historiques des mesures, des alarmes,...), etc..
Configurations Superviseur	Entrée	Pour s'adapter à l'environnement et faire varier la sensibilité des mesures, il est possible de reconfigurer certains paramètres clés du D.E.C comme les seuils minima et maxima des différents comparateurs ou encore des valeurs de référence. Ces paramètres permettent de délimiter des zones de fonctionnement nominales, de vérifier la fiabilité des capteurs et de l'alimentation, et enfin d'autoriser ou non la mémorisation d'une mesure prélevée (si la dernière mesure a très peu variée par rapport à la précédente, il est inutile de la conserver en mémoire, ce qui évite la saturation de celle-ci).
Environnement (accélérométrie, température, champ électromagnétique)	Entrée	L'Environnement du D.E.C représente la partie mécanique stratégique à observer. Des mesures d'accélérométrie, de température et du champ électromagnétique vont donc être effectuées sur cet environnement et permettront d'étudier l'évolution du système dans le temps selon les variations des conditions environnementales.

Désignation	Type	Description
Informations Unité de contrôle	Sortie	Ce sont des informations transmises à l'unité de contrôle indiquant l'état du système, le mode courant, un historique de mesures, le résultat de tests et de commandes exécutées.
Messages d'Alarme	Sortie	Pour indiquer un fonctionnement anormal du système observé, le D.E.C envoie des messages d'alarmes à l'unité de contrôle en lui fournissant toutes les données nécessaires pour traiter les problèmes apparus ou réinitialiser le système. Ces défauts dans le fonctionnement du système peuvent avoir plusieurs origines : soit des résultats de tests indiquent la présence d'anomalies dans le circuit surveillé ; soit la détection d'un fonctionnement hors des limites nominales (les mesures fournies par les capteurs en entrées sont soit supérieures, soit inférieures et ne correspondent plus aux paramètres de configuration), soit une commande n'est pas reconnue par le système superviseur, soit le système est dans un état bloqué.

4.3.1 Partition Fonctionnelle

Le Micro Système D.E.C possède huit organes autonomes dans leur fonctionnement mais qui sont coordonnés et surveillés par un organe superviseur integré chargé de surveiller l'opération correcte de chacun des organes. Le superviseur se charge aussi de matérialiser la stratégie adequate en cas de présence d'anomalies de fonctionnement du D.E.C.

La coordination des organes est réalisée en deux groupes mutuellement exclusifs :

- Le premier groupe est coordonné et contrôlé quand l'avion est en vol, qu'il existe au moins un capteur en bon état et que l'alimentation est suffisante. Les organes sont Mesurer, Conditionner, Traiter, Mémoriser et Tester. Ils se chargent d'acquérir des mesures externes, de les traiter, de les analyser

et de les mémoriser si elles sont pertinentes (significatives). Ceci correspond au mode de fonctionnement Permanent (Normal ou Intermediaire). Si aucun des capteurs est en bon état ou que le niveau d'alimentation n'est suffisant, le superviseur re programme les organes du Micro Système pour enter le mode Dégradé. L'alimentation est coupé aux organes chargés de prendre, traiter et analyser les mesures externes.

- Le deuxième groupe correspond à la réception et à la transmission des informations et des messages quand l'avion est au sol. Les organes correspondants sont Mémoriser et Communiquer.

L'architecture générique d'un D.E.C (voir figure 4.3) se compose de neuf organes qui sont :

- Polariser
- Mesurer
- Conditionner
- Traiter
- Mémoriser
- Communiquer
- "Universal Time"
- Superviser
- Tester

Bloc Polariser

Il est chargé de capter, de réguler et de distribuer l'énergie aux autres blocs conformant le D.E.C. Le superviseur fait la demande d'alimentation des blocs à activer, en lui indiquant de comencer à capter l'énergie. L' organe polariser se mantiendra actif fournissant l'énergie nécessaire jusqu'a que le niveau d'alimentation atteigne un niveau trop bas pour continuer une opération correcte. Quand cet événement arrive, l' organe polariser envoie un signal au superviseur en lui indiquant que l'énergie n'est pas suffisante pour mantenir allumés tous

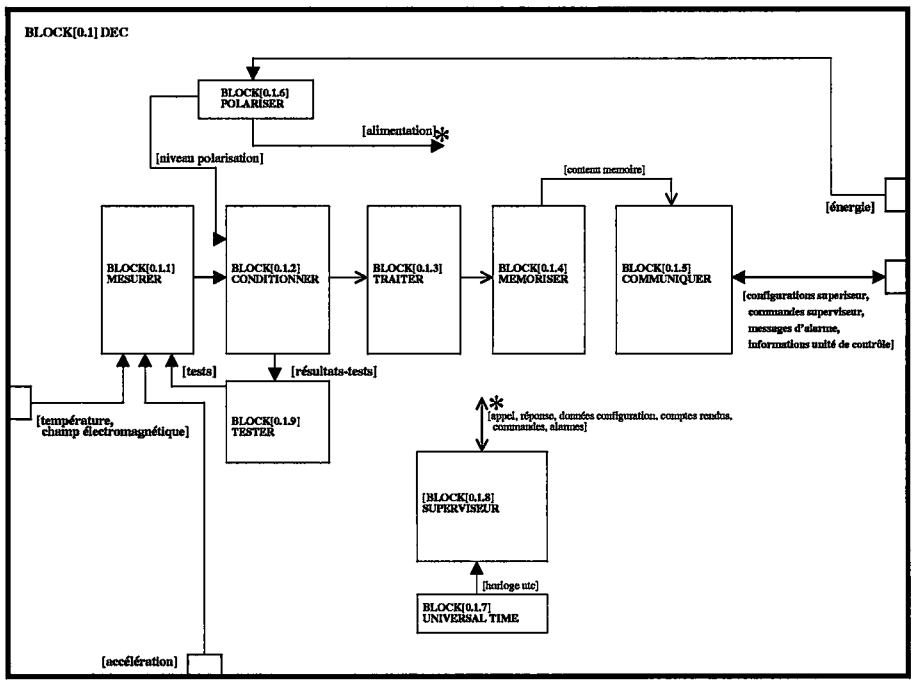

FIG. 4.3 – *Architecture Générique d'un D.E.C.*

les organes. Ceci correspond à une détection d'un symptôme de défaillance et le superviseur se charge de changer le mode de fonctionnement au mode dégradé.

Le bloc Polariser doit fournir l'énergie électrique nécessaire au bon fonctionnement du dispositif. Cependant, tous les circuits ne requièrent pas forcément la même valeur de tension d'alimentation, il faudra donc prévoir le cas échéant des convertisseurs abaisseurs ou élévateurs. De plus, il est impératif que les circuits soient alimentés sous une tension constante, notamment les accéléromètres car leur précision est proportionnelle à la tension d'alimentation. Pour cela, il faudra insérer un régulateur de tension entre la source et les circuits. Ce bloc est donc constitué de trois sous blocs clés (voir figure 4.4):

- Bloc alimenter (ex: un accumulateur),
- Bloc conditionner (ex: un régulateur, les convertisseurs abaisseurs / élévateurs, micro relais),
- Bloc actionner (ex: micro contacts).

dont les principaux paramètres sont les tensions d'alimentation des différents circuits et la consommation énergétique totale (Ampère.heure).

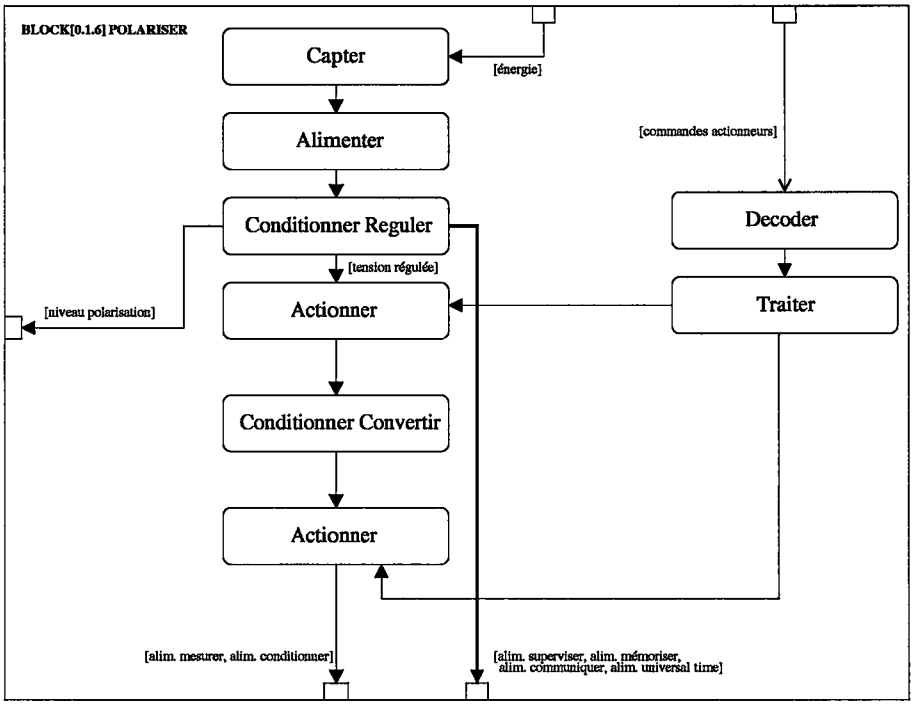

FIG. 4.4 – *Architecture du Bloc Polariser*

Désignation	Type	Description
Energie (RF/Optique)	Entrée	Cette énergie provient de l'environnement extérieur par radiofréquence ou optique. Elle va être adaptée par l'intermédiaire de convertisseurs et répartie pour alimenter correctement les différents blocs connectés selon les directives du bloc Superviseur.
Commandes Micro Relais	Entrée	Ces commandes proviennent du bloc Superviser qui, en de différents paramètres (Mode courant, Messages d'Alarmes, Commandes Superviseur) va configurer l'état on/off des contacts des micro relais et ainsi imposer l'état de fonctionnement des différents blocs à alimenter.
Alimentation Mesurer	Sortie	Alimentation du bloc Mesurer
Alimentation Superviser	Sortie	Alimentation du bloc Superviser
Alimentation Emettre / Recevoir	Sortie	Alimentation du bloc Emettre / Recevoir
Test Alimentation	Sortie	Indique le niveau d'énergie dans la micro pile. Cette information va être envoyée au bloc Superviser qui, selon les seuils, pourra indiquer si ce niveau est acceptable ou non.

Bloc Mesurer

Cet organe comporte l'ensemble des capteurs et leurs circuits de polarisation. Les capteurs fournissent à leur sortie une valeur électrique (tension ou courant) proportionnelle à la quantité physique mesurée en entrée. Les capteurs peuvent recevoir les signaux physiques de deux sources différentes. L'une est la mesure externe quand le D.E.C est en mode de fonctionnement permanent. L'autre correspond aux valeurs induites pour les actionneurs du bloc de test et commandés par le diagnostic.

Le superviseur se charge d'envoyer l'ordre de début, il envoie un signal d'alimentation et le signal de début de temporisation du mode d'opération. Si le niveau d'alimentation n'est pas suffisant, l'organe envoie un signal de détection au superviseur indiquant la condition de défaillance.

L'organe peut finir son éxecution au plus tard quand le temporisateur arrive à sa fin. Le superviseur a la capacité d'achever l'éxecution de l'organe quand le premier le décide.

Actuellement et plus encore à l'avenir, la tendance est à l'intégration de composants électroniques permettant une mise en forme primaire du signal au plus près de la partie mécanique du capteur (Capteurs Intelligents). Dès lors, ces mêmes capteurs fournissent un signal électrique avec un rapport signal / bruit de plus en plus élevé, augmentant ainsi la facilité et la qualité de traitement analogique qui s'ensuit ainsi que la conversion en signal numérique. Toutefois, la précision des mesures pouvant parfois dépendre de la valeur de la tension d'alimentation, il sera nécessaire d'utiliser des convertisseurs élévateurs de tension lorsque la précision sera un paramètre très critique (voir figure 4.5).

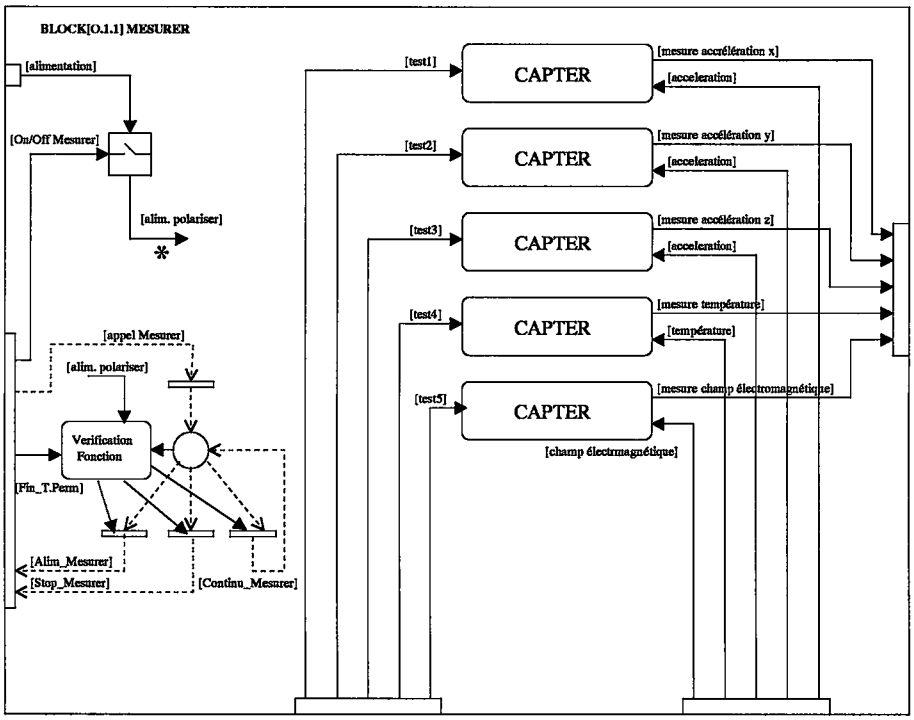

FIG. 4.5 – *Architecture du Bloc Mesurer*

Désignation	Type	Description
Alimentation Mesurer	Entrée	Alimentation en tension des différents capteurs.
Environnement (accélérométrie, température, champ électromagnétique)	Entrée	Ce sont des mesures d'accélération, de température et du champ électromagnétique auxquels sont soumis les systèmes observés et elles seront transmises sous forme analogique au bloc Superviser.
Tests Capteurs	Entrée	Un des points forts de ce Micro Système est l'autonomie, c'est à dire la possibilité d'adaptation (changement de mode, passage en mode dégradé, etc..) en cas d'événements anormaux. Pour cela, il effectue des tests (en permanence ou sur demande) sur les micro capteurs afin de vérifier le bon fonctionnement de ceux-ci. Le résultat de ces tests sera envoyé au bloc Superviser qui pourra dès lors établir un diagnostic sur l'état du système, prendre les mesures appropriées et le cas échéant, envoyer des messages d'alarme.
Donnée analogique 1, ..., Donnée analogique 5	Sortie	Ces données analogiques représentent des tensions proportionnelles aux grandeurs physiques mesurées (accélération, température, champ magnétique) et le résultat d'autotests

Bloc Conditionner

De même que pour l'organe mesurer, le superviseur se charge d'envoyer les données de configuration et d'initilisation. Des lors que le niveau d'alimentation sera approprié et que la temporisation du mode courant sera terminée, l'organe démarrera son exécution. Si le niveau d'alimentation est sous le seuil minimal ou si le temporisateur du mode courant arrive à sa fin, l'organe signale cet événement au superviseur.

L'organe conditionner requiert l'actualisation de toutes les valeurs des me-

sures externes prises. Un jeton provenant de l'organe traiter est chargé d'actualiser les valeurs de configuration du multiplexeur de mesures chaque fois que une valeur sur un capteur a été prise.

La tâche principale de ce bloc va être la transformation de données analogiques en informations directement exploitables par le bloc traiter. Pour cela, elle dispose des bloc spécialisés en traiter et conditionner : amplificateurs en tension, échantillonneurs bloqueurs, filtres, un multiplexeur analogique, un convertisseur analogique / numérique et un démultiplexeur numérique. (voir figure 4.6).

Une mesure analogique est prélevée sur l'environnement toutes les 100 ms minimum. Elle est amplifiée, filtrée et envoyée à l'entrée d'un multiplexeur analogique. Les mesures de l'accélération (selon les trois axes), de la température et du champ électromagnétique constitueront les 5 entrées de ce multiplexeur. La re-direction d'une de ces entrées vers la sortie sera commandée par une combinaison de trois bits de contrôle envoyée par le bloc Traiter. Lorsqu'une mesure est sélectionnée, elle traverse un convertisseur analogique / numérique et arrive à l'entrée d'un dé-multiplexeur. Ce dé-multiplexeur, commandé par le bloc Superviseur, permet de rediriger la mesure vers le bloc traiter.

Bloc Traiter

Cet organe traite et vérifie si la mesure prise doit ou ne doit pas être mémorisée.

Le superviseur envoie les signaux d'alimentation, de début de temporisation et le jeton de demande pour que l'organe démarre son opération. L'organe a deux signaux de configuration pour initier son fonctionnement : l'un pour débuter la prise de mesures et l'autre pour continuer la prise des mesures après la détection d'un symptôme de défaillance et que le diagnostic ait trouvé que la mésure vient effectivement de l'environnement externe. L'organe détecte trois types d'événements : la fin du temporisateur d'opération (cette valeur peut être configurée par le superviseur) , l'avion est au repos, la mesure dépasse les seuils de fonctionnement du capteur (saturation) et la défaillance de l'alimentation (voir figure 3.5).

Pour éviter la saturation prématurée du bloc mémorisation, le bloc traiter va effectuer des opérations sur la mesure qu'il reçoit pour vérifier si la variation par rapport à la donnée précédente est suffisamment significative pour être mémorisée. Si c'est le cas, la donnée est datée avec une vignette de temps universel

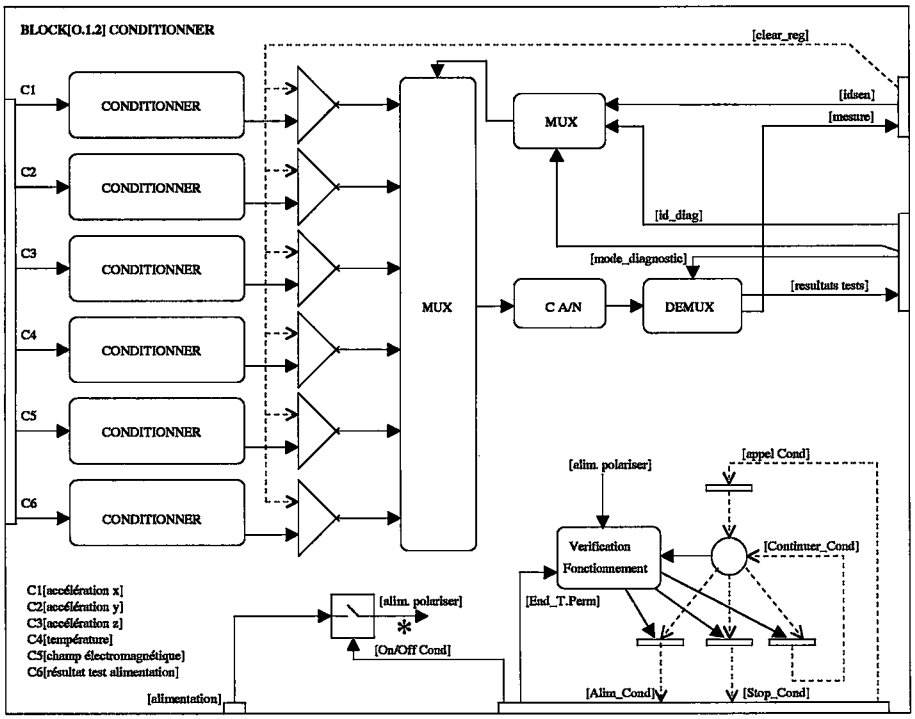

FIG. 4.6 – *Architecture du Bloc Conditionner*

et ensuite transférée au bloc mémorisation, sinon elle est perdue (mode normal / acquisitions de mesures). Lorsque ces données correspondent à des résultats de tests, elles pourront être redirigées vers le bloc de diagnostic.

Bloc Mémoriser

Son rôle est de mémoriser les mesures pertinentes et les alarmes, même en absence d'une alimentation électrique fiable. Les mesures sont recues du bloc Traiter et elles peuvent être envoyées vers une pile de registres de type "First In First Out" pouvant recevoir une dizaine de mesures ou vers une mémoire vive de type RAM qui conservera ces mesures afin d'établir des historiques de mesures.

Bloc Superviseur

Le bloc *Superviseur* est l'élément principal du système. Il va gérer et surveiller le fonctionnement de l'ensemble et communiquer avec l'extérieur. Son rôle est d'assurer que les signaux fournis par les différents capteurs vont être convertis en informations exploitables permettant de mettre à jour les informations nécessaires à la maintenance préventive et prédictive de la structure mécanique de l'avion.

En outre, le bloc Superviser sera suffisamment autonome pour prendre les décisions qui s'imposent en cas d'événements anormaux. En effet, il pourra effectuer des tests sur l'ensemble des éléments composant le système (capteurs, niveaux d'énergie, chaînes d'acquisition et de traitement, interface de communication, etc.), il sera capable d'établir un diagnostic complet sur l'état général, continuer à fonctionner même en cas de perte d'un ou plusieurs capteurs, décider de passer en mode veille pour optimiser la consommation d'énergie ou en mode dégradé si le niveau est trop faible.

Bloc Communiquer

Le bloc Communiquer permet d'échanger des informations en série avec l'extérieur (voir figure 4.7). C'est aussi un bloc spécifique car il dépend essentiellement du médium de communication que l'on souhaite utiliser. Plusieurs possibilités :

- communication filaire : électrique / fibre optique

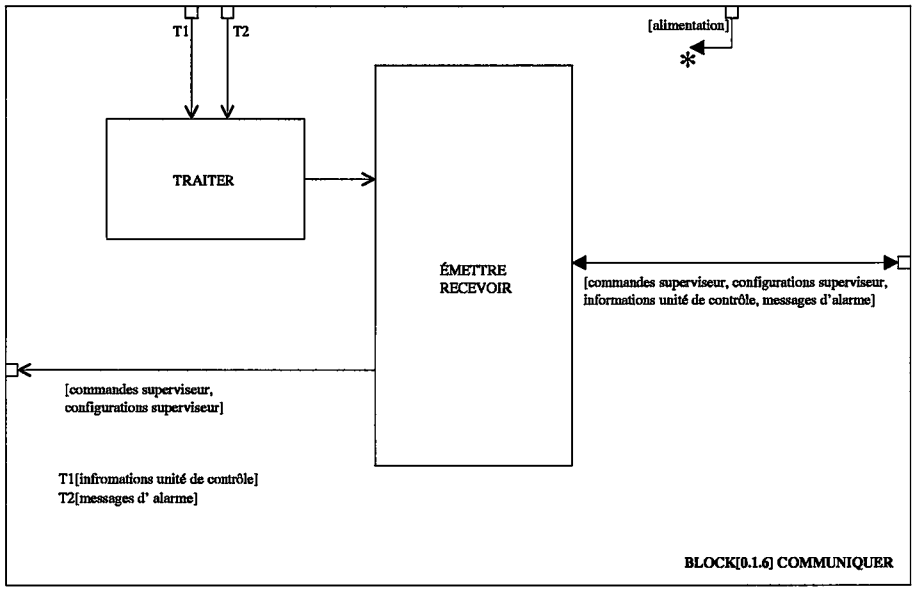

FIG. 4.7 – *Architecture du Bloc Communiquer*

– sans fil : radiofréquence / optique

Ainsi, le choix du médium de communication se fera en fonction de différents critères : facilité de mise en oeuvre d'un réseau de D.E.C., compatibilité électromagnétique, distances de communication, présence d'obstacles, pertes,..., mais le passage de l'un à l'autre doit rester souple.

Émission Concernant le bloc Mémorisation, plusieurs conditions permettent de le vider vers l'unité de contrôle :

– lorsqu'il arrive à saturation et seulement si l'avion est au sol, toutes les informations pourront être transférées, sur demande de l'unité de contrôle ;

– lors de l'envoi d'un message d'alarme,

Cependant, avant d'être transmises, les données passent par un bloc traiter (protocole d'émission) qui associe à ces informations des champs de bits particuliers pour leurs mises en forme (code correcteur d'erreurs, code d 'identification, etc.).

Réception L'unité de contrôle gère le fonctionnement global du D.E.C. quand l'avion est au sol. Le D.E.C est autonome et ne communique pas pendant

que l'avion est en vol. Dans le premier cas, l'unité de contrôle peut à tout instant interrompre les activités en cours du D.E.C. en lui demandant d'effectuer des opérations particulières. Ces opérations peuvent être:

- séquences de tests avec le renvoi des résultats correspondants,
- changement de mode de fonctionnement,
- renvoi d'historiques (mesures et/ou alarmes),
- programmation de paramètres (seuils, horloges, etc.),
- réinitialisation.

Ainsi, le D.E.C. reçoit des commandes qui vont traverser le bloc communiquer avant d'arriver à un décodeur de messages à l'intérieur du bloc Superviseur. A partir de là, le bloc Superviseur envoi les séquences de commandes qui doivent être exécutées par le D.E.C.

Bloc "Universal Time"

Son rôle est de recevoir, maintenir et fournir sur demande du bloc superviseur l'UTC ("Universal Time Coordinated ") [66]. Il sera capable de corriger les déviations de l'UTC par le fait des retards de communication.

Bloc Tester

Son rôle est d'éffectuer des tests sur les organes du D.E.C pour prendre des informations et connaître l'état général du système (voir figure 4.8).

Un diagnostic comporte des informations concernant l'état des capteurs et le mode de fonctionnement courant. Deux cas peuvent provoquer l'établissement d'un diagnostic:

- sur requête de l'unité de contrôle qui souhaite avoir une vue générale sur le fonctionnement du dispositif,
- suite à la détection d'une mesure hors du domaine nominal de fonctionnement.

Concernant le deuxième cas, le bloc Superviseur reçoit un signal provenant du bloc Traiter lui demandant d'effectuer un diagnostic. Dès lors, il va faire un

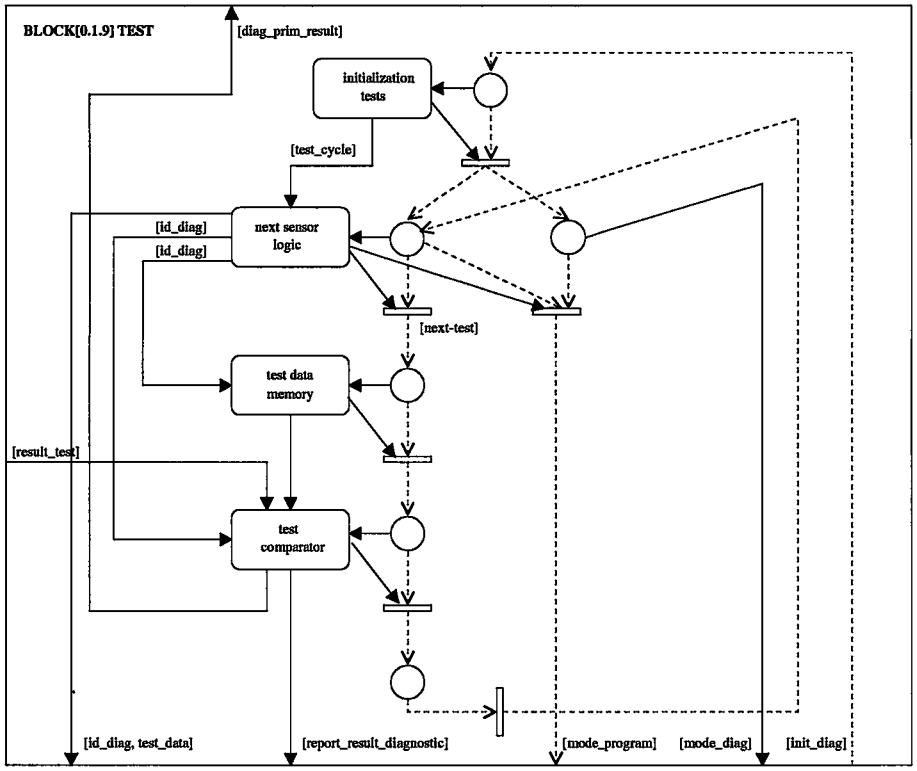

FIG. 4.8 – *Architecture du Bloc Test*

appel au bloc Test pour programmer le bloc Conditionner et rediriger le signal d'entrée vers la sortie correspondant au test. Ensuite, il va envoyer des séquences de tests aux capteurs. Les diverses réponses se retrouvent séquentiellement à l'entrée du comparateur 'Diagnostic' qui va les comparer à des valeurs prédéterminées. Par conséquent, la comparaison des réponses et des valeurs prédéterminées permettra d'indiquer si tel ou tel capteur est en parfait état ou s'il est détérioré.

4.3.2 Définitions des Modes de Fonctionnement

Le D.E.C peut fonctionner selon plusieurs modes différents :

- Mode Normal :
  - mode permanent normal
  - mode veille normal

- Mode Intermédiaire:
  - mode permanent intermédiaire
  - mode veille intermédiaire
- Mode Dégradé

Mode Normal

Ce mode normal comporte trois sous modes qui caractérisent un fonctionnement normal du système. Ici, toutes les parties sensibles telles que les capteurs ou l'alimentation sont en parfait état de marche.

Fonctionnement Permanent Normal: Le superviseur intégré envoie aux micro relais une commande autorisant l'alimentation de tous les blocs composant le circuit. Ce mode est caractérisé par le prélèvement en permanence de mesures sur l'environnement observé. Ces mesures vont ensuite être envoyées vers le Superviseur pour y être traitées.

Fonctionnement Veille Normal: C'est le mode de fonctionnement par défaut, c'est à dire que le D.E.C. se trouve dans ce mode après une réinitialisation du dispositif. Dans ce type de fonctionnement, seuls le Superviseur et le bloc communiquer sont alimentés; tous les autres sont au repos. Lorsque l'avion est à l'arrêt, la prise de mesures n'est pas nécessaire, le D.E.C. se place en mode veille (le bloc programmation envoie aux micro relais la commande autorisant ce passage). Sachant que le bloc communiquer reste alimenté en permanence, il attend l'arrivée de commandes éventuelles qui informeraient d'un changement de mode de fonctionnement.

Mode Intermédiaire

Ce mode est équivalent au mode normal. Cependant, ici, le niveau d'énergie est aussi convenable mais il y a au minimum un capteur qui fonctionne correctement et un capteur détérioré. Ce mode est atteint après avoir détecté un dysfonctionnement sur un ou plusieurs capteurs. Avant de poursuivre les mesures, il faut localiser la ou les entrées du multiplexeur correspondant aux capteurs défectueux pour ne plus les sélectionner. Dans ce mode intermédiaire,

il y a aussi trois sous-modes qui ont déjà été décrits précédemment :

- mode permanent intermédiaire
- mode veille intermédiaire

Ainsi, la différence entre ces sous-modes et ceux appartenant au mode normal est le nombre de capteurs détériorés.

Mode Dégradé

Le D.E.C. se trouve dans ce mode dégradé à la suite de tests qui indiquent que le niveau d'énergie disponible est insuffisant pour assurer un fonctionnement nominal du système et/ou une défaillance des capteurs (détériorés ou fournissent des mesures non représentatives de la réalité). Si seul le niveau d'énergie est en cause, il est possible de recharger le D.E.C. et de le réinitialiser.

Réinitialisation

Comme son nom l'indique, ce mode permet de réinitialiser le système lors d'une première utilisation, ou à la sortie d'un mode dégradé si seulement le niveau d'alimentation était trop bas et que tous les capteurs fonctionnent normalement, ou encore sur demande de l'unité de contrôle. La procédure de réinitialisation est la suivante :

- les différentes mémoires vives (utilisées pour des calculs intermédiaires) du D.E.C. sont remises à zéro,
- le Superviseur récupère dans sa mémoire les différentes valeurs de seuils pour tous les montages comparateurs,
- le bloc programmation envoie une commande aux microrelais permettant le passage en mode veille normal du D.E.C. afin de patienter jusqu'à la réception d'une nouvelle commande provenant de l'unité de contrôle.

Des que le mode veille est atteint, le dispositif est à nouveau opérationnel.

4.3.3 Partition Discret - Continu

En ayant comme souci un prototypage rapide, une partition naturelle entre continu et discret a été proposée. Les blocs mesurer, alimenter et la partie analogique des blocs communiquer et conditionner, ont été modélisés en temps

continu avec des signaux continus. Les blocs superviseur, traiter, mémoriser et les parties numériques des blocs conditionner et communiquer ont été modélisées par événements discrets avec des signaux binaires. Nous avons pris les blocs disponibles dans la bibliothèque, et après une paramétrage adéquat aux fonctionnalités spécifiques, un prototype virtuel a été généré.

Par la suite, nous allons présenter la conception détaillée des blocs superviseur et traiter, pour montrer notre contribution à la conception des Micro Systèmes. Nous allons apporter les avantages d'une conception basée sur des circuits asynchrones, telles que la basse consommation de puissance électrique, la basse émission de bruit électromagnétique et l'adaptabilité aux variations des paramètres physiques. Le prototype virtuel complet du D.E.C est montré en annexe.

4.4 Conception de l'architecture des blocs Traiter et Superviseur

4.4.1 Architecture du bloc Traiter

Ce bloc comporte trois sous blocs de traitement distincts qui correspondent :

- à un fonctionnement normal d'acquisition de mesures (bloc Comparateur),
- à l'activité de dater les mesures pertinentes avant de les envoyer au bloc Mémorisation.

(voir figure 4.9)

Architecture du bloc "Processing"

(voir figure 4.10)

En fonctionnement normal, la mesure va traverser un premier comparateur (mémorisation) qui indiquera s'il est nécessaire de mémoriser cette mesure. Le principe de conservation de la dernière mesure est le suivant : la donnée en cours de traitement est mémorisée seulement si la variation par rapport à la valeur précédente dépasse une valeur relative ΔM (reconfiguration) afin d'éviter une saturation prématurée de la mémoire du Superviseur par l'encombrement de mesures inutiles et de ne conserver que celles qui sont pertinentes. Par la suite,

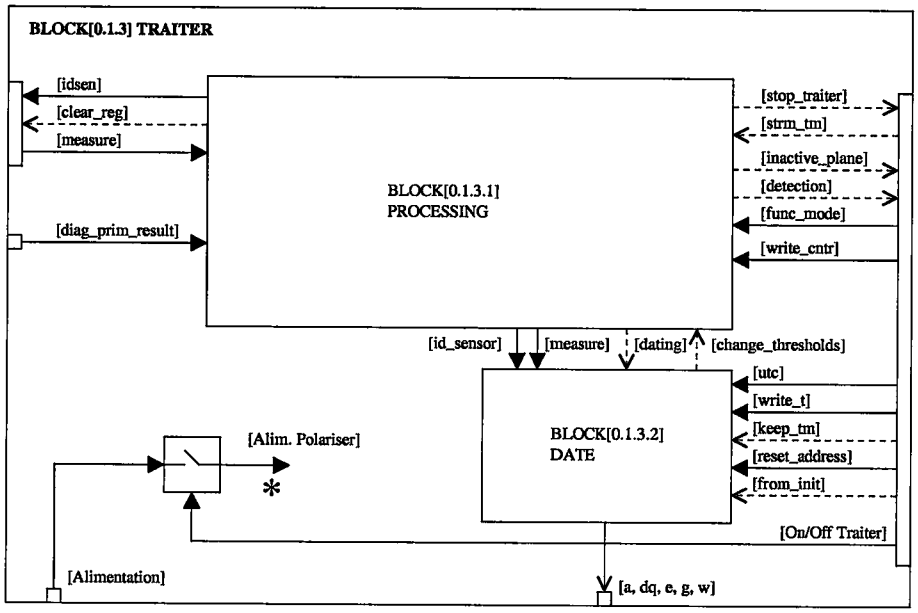

FIG. 4.9 – *Architecture du Bloc Traiter*

si cette donnée est mémorisable, elle traverse un deuxième comparateur (fonctionnement) dont le rôle est de déterminer si la mesure appartient à un domaine nominal de fonctionnement ou est en dehors de celui-ci. Si la mesure est hors du domaine nominal, le comparateur envoie un signal au bloc Superviseur pour lui demander d'établir un diagnostic sur l'état du système afin de vérifier que la mesure retranscrit correctement un événement particulier et n'est pas due à une défaillance d'un des capteurs. Sinon, si la donnée indique un fonctionnement nominal, elle est envoyée au bloc Dater

Architecture du bloc Dater

Dans le bloc Dater, une vignette temporelle (bloc Dater) est adjointe à la mesure permettant d'indiquer sa date d'acquisition. A partir de là, elle est envoyée vers le bloc mémorisation. (voir figure 4.11)

4.4.2 Architecture du bloc Superviseur

Nous proposons d'utiliser l'architecture générique de bloc superviseur définie au chapitre précédent. Ici, nous allons personnaliser les blocs de détection, diagnostic et programmation aux spécificités de l'application en cours de conception

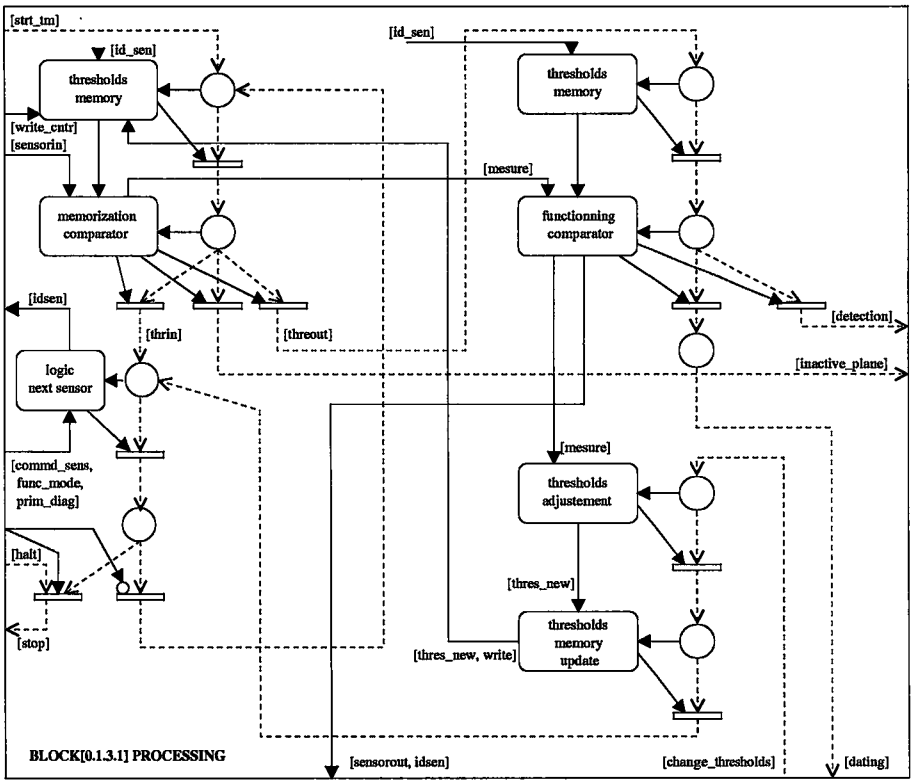

FIG. 4.10 – *Architecture du Bloc "Processing"*

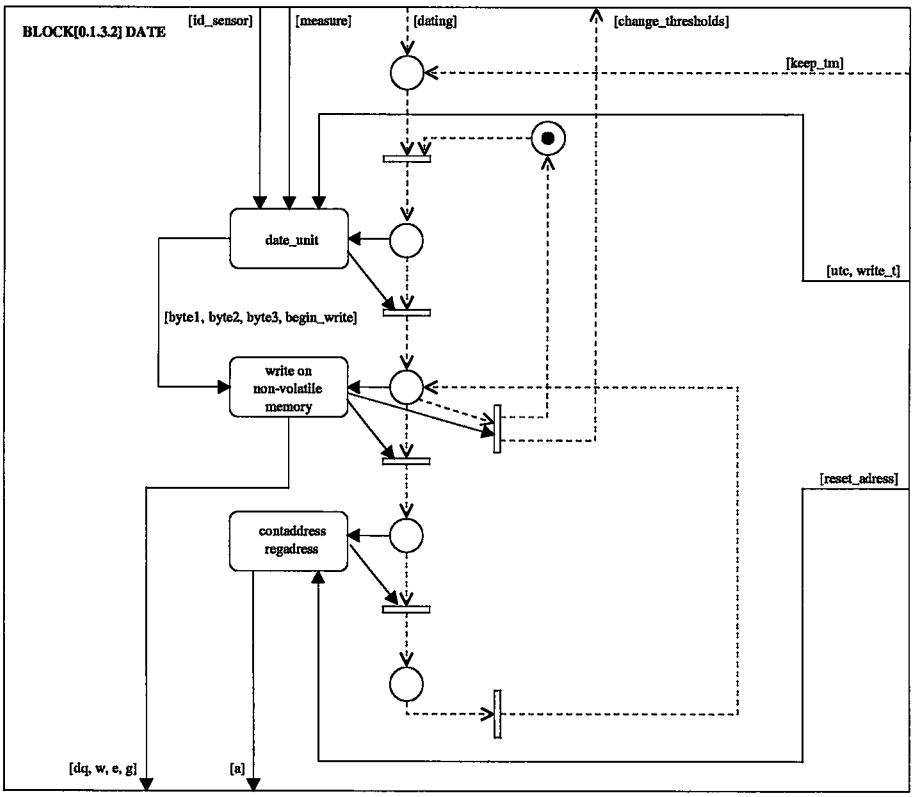

FIG. 4.11 – *Architecture du Bloc Dater*

(voir figure 4.12).

Une logique d'analyse de diagnostic intégrée dans le Superviseur indiquera le mode de fonctionnement à reprogrammer (s'il est différent du précédent). Parmi tous les modes de fonctionnement possibles, le Superviseur ne peut gérer que le passage vers l'un des quatre modes suivants :

- mode normal,
- mode intermédiaire,
- mode dégradé,
- mode diagnostic.

Les blocs réinitialisation et programmation sont commandés par l'unité de contrôle. A chacun des blocs correspond une commande particulière, celle-ci est envoyée par le Superviseur vers les micro relais et va permettre de configurer les différents contacts et donc les différents convertisseurs de telle sorte que le mode de fonctionnement désiré soit atteint.

Bloc Programmation Le bloc de Programmation permet de changer de mode de fonctionnement courant (résultant d'une nouvelle commande ou d'un diagnostic), d'exécuter des séquences de tests et renvoyer les résultats correspondants, de transmettre des historiques (mesures, anomalies, etc.), de reconfigurer les différents seuils des montages comparateurs et enfin de reprogrammer l'horloge interne (temps universel). Deux possibilités pour sortir de ce bloc :

- l'unité de contrôle a transmis une commande correspondant au changement de mode de fonctionnement,
- le D.E.C. revient dans le mode précédant l'interruption dès la fin du renvoi des informations (résultats tests, historiques, etc.) vers l'unité de contrôle.

Bloc Diagnostic L'activation du bloc diagnostic peut être effectuée de trois façons :

- sur demande : l'unité de contrôle veut avoir un diagnostic sur l'état général du système,
- cycliquement : des tests sont effectués sur l'ensemble des parties sensibles pour assurer une fiabilité permanente,

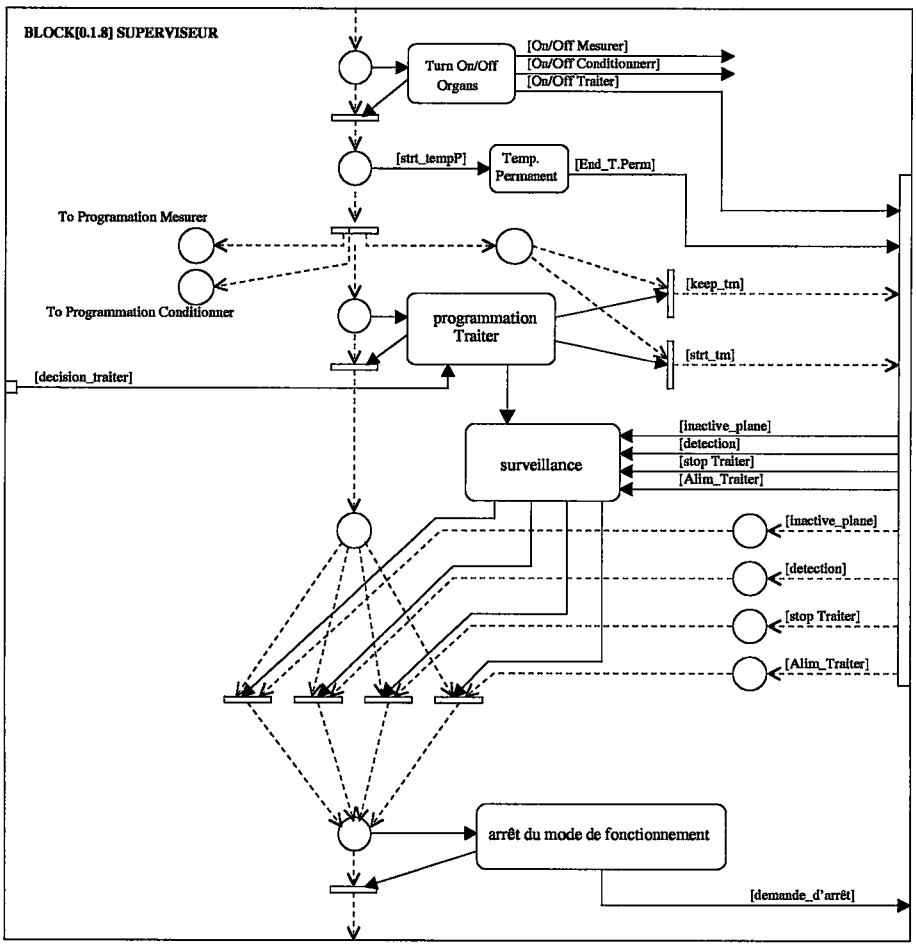

FIG. 4.12 – *Architecture Interne du Bloc Superviseur*

- ponctuellement : à la suite de la détection d'une mesure en dehors d'un domaine de fonctionnement nominal.

Ce diagnostic comprendra des résultats de tests (capteurs et alimentation) ainsi que le mode de fonctionnement courant. Si le diagnostic est correct, le D.E.C. revient dans un mode de fonctionnement précédant le passage en mode diagnostic. Sinon, si des anomalies sont détectées, des messages d'alarme avec le diagnostic pourront être envoyés à l'unité de contrôle et le Superviseur pourra prendre l'initiative de passer :

- en mode dégradé : si tous les capteurs sont détériorés ou si le niveau d'alimentation est trop bas,
- en mode intermédiaire : si le niveau d'énergie est convenable et si au moins un capteur fonctionne correctement.

Désignation	Type	Description
Alimentation Superviser / Alimentation Veille Superviser	Entrée	Alimentation en tension des différents circuits
Donnée analogique 1, ..., Donnée analogique 5	Entrées	Données analogiques provenant des accéléromètres, du capteur de température et du capteur de champ électromagnétique. Ces données sont proportionnelles aux grandeurs physiques mesurées et peuvent aussi être le résultat d'autotests effectués sur les différents capteurs.
Commandes Superviseur	Entrée	Le DEC est un dispositif autonome, autrement dit, il pourra gérer lui-même son fonctionnement. Cependant, l'Unité de Contrôle peut à tout moment interrompre les activités du DEC en lui envoyant des commandes pour changer le mode de fonctionnement courant, générer des tests et renvoyer les résultats correspondants afin de vérifier le bon déroulement des opérations, obtenir des informations concernant l'environnement observé (historique des mesures ou des événements).
Configurations Superviseur	Entrée	Pour s'adapter à l'environnement et faire varier la sensibilité des mesures, il est possible de reconfigurer certains paramètres clés du DEC comme les seuils des différents comparateurs. Ces seuils permettent de délimiter des zones de fonctionnement nominales, de vérifier la fiabilité des capteurs et de l'alimentation, et enfin d'autoriser ou non la mémorisation d'une mesure prélevée (si la dernière mesure a très peu variée par rapport à la précédente, il est inutile de la conserver en mémoire, ce qui évite la saturation de celle-ci).
Test Alimentation	Entrée	Ce test indique le niveau d'énergie disponible dans la micro pile. Pour parer à tout événement anormal et établir un diagnostic complet, il est essentiel de connaître ce niveau d'énergie.

Désignation	Type	Description
Prises / Séquencement des mesures	Interne	Cette commande provient du bloc Commander Surveiller et permettra de contrôler la prise ainsi que le séquencement des mesures. Trois principales configurations sont possibles : en mode de fonctionnement normal, les cinq mesures physiques sur les six entrées disponibles seront transférées l'une après l'autre à la chaîne de mémorisation des informations (Séquencement Mesures) ; en mode commandé, il est possible de n'étudier qu'un seul paramètre (accélération ou température ou champ électromagnétique). Ainsi, pendant une durée limitée, la même entrée sera redirigée vers la sortie (Prise Mesures) ; en mode diagnostic, toutes les entrées seront sélectionnées de manière séquentielle.
Mode Normal / Diagnostic	Interne	Les informations numériques qui arrivent à l'entrée du démultiplexeur correspondent soit à des mesures caractérisant l'environnement du système, soit à des résultats de tests effectués sur les capteurs et l'alimentation. Par conséquent, selon le mode de fonctionnement, le traitement de ces données sera différent et la commande Mode Normal / Diagnostic provenant du bloc Commander Surveiller permettra en contrôlant le démultiplexeur de les rediriger vers la sortie adéquate.
Résultats Tests	Interne	Ces informations résultent de tests effectués d'une part, sur les capteurs pour vérifier que les valeurs analogiques transmises par ceux-ci sont fiables et d'autre part, sur le niveau d'alimentation disponible dans la micro pile.

Désignation	Type	Description
Paramètres Configurations	Interne	Dans le cas présent, ces paramètres seront des seuils relatifs délimitant une zone de sensibilité autour de la mesure reçue. Si la donnée en cours de traitement appartient à cette zone de sensibilité, cela implique que la variation par rapport à la donnée précédant n'est pas significative et qu'il est alors inutile de la conserver en mémoire. Par conséquent, elle sera perdue. Sinon, si elle est en dehors de cette zone de sensibilité, il faudra la transmettre au bloc Commander Surveiller afin d'être mémorisée.
Mesures	Interne	Ce sont des informations numériques qu'il faudra conserver en mémoire et qui représentent l'évolution des conditions environnementales auxquels sont soumis les systèmes observés. A ce niveau, elles vont subir un nouveau traitement qui va permettre de détecter un événement anormal dans l'environnement du système.
Informations Unité de contrôle	Sortie	Ces informations concernent le fonctionnement global du système observé et résultent : de mesures prélevées (en cours ou historiques), de tests effectués sur les capteurs et l'alimentation, de reconfiguration des paramètres (seuils de comparaison), de commandes exécutées, de renseignements sur un fonctionnement normal ou anormal ainsi que sur le mode de fonctionnement courant du DEC (permanent, échantillonné, commandé, veille, dégradé). Grâce à ces informations, l'unité de contrôle peut ainsi avoir une vue globale sur le fonctionnement du DEC ainsi que sur l'évolution des conditions extérieures sous lesquelles évolue le système observé.

Désignation	Type	Description
Messages d'Alarme	Sortie	Pour indiquer un fonctionnement anormal du système observé, le DEC envoie des Messages d'Alarmes à l'unité de contrôle en lui fournissant toutes les données nécessaires pour traiter les problèmes apparus ou réinitialiser le système. Ces défauts dans le fonctionnement du système peuvent avoir plusieurs origines: des résultats de tests indiquent la présence d'anomalies dans le circuit surveillé; détection d'un fonctionnement hors des limites nominales (les mesures fournies par les capteurs en entrées sont soit supérieures, soit inférieures et ne correspondent plus aux paramètres de configuration); une commande n'est pas reconnue par le système superviseur ; le système est dans un état bloqué.
Commandes Micro Relais	Sortie	Ces commandes, qui dépendent de différents paramètres (Mode courant, Messages d'Alarmes, Commandes Superviseur), sont envoyées au module Alimenter et vont permettre de configurer l'état on/off des contacts des micro relais et ainsi imposer l'état de fonctionnement des différents modules à alimenter.
Tests Capteurs	Sortie	Ce sont des signaux transmis à des actionneurs appartenant au Module Prendre des Mesures et qui vont exciter les capteurs. Les réponses de ceux-ci seront analysées à travers une chaîne de diagnostic qui permettra d'indiquer si les capteurs fonctionnent correctement ou si certains d'entre eux sont endommagés.

4.5 Validation Structurelle et Vérification Fonctionnelles

La validation architecturale a deux volets différents :

- Validation par règles de construction (connexions orientées interfaces, connexions correctes, signalisation approprie indépendant de la vitesse (circuits asynchrones), contrôle d'accès, etc).
- validation par simulation du réseau de Petri associé ;

La vérification fonctionnelle a été faite par simulation comportementale des descriptions architecturales en VHDL-AMS au niveau RTL. Nous avons utilisé le logiciel Anacad de Mentor.

4.5.1 Modèles RTL

Nous avons utilisé la codification montré par la figure 4.13, pour synthétiser les architectures décrites à un niveau RTL. Ceci permet d'engager par la suite la conception aval.

Une description de plus bas niveau a été testée et vérifiée. Un exemple est montré par la figure 4.14.

4.5.2 Librairie des blocs micro architecturaux

Nous avons développé une librairie des blocs micro architecturaux, pour montrer l'allocation directe des descriptions HiLeS à des éléments physiques asynchrones. Le choix d'une réalisation par ASIC amène des avantages au niveau du coût et de l'intégration monolithique et hétérogène. Les blocs micro architecturaux on été développés en technologie AMS 0.8 μm. La librairie a été enrichie par les cellules suivantes :

- C de Muller,
- de contrôle de l'appel réponse à signalisation de 4 phases (retour à zero),
- sélecteurs,
- bascule transparente,
- élément de décision et d'attente,

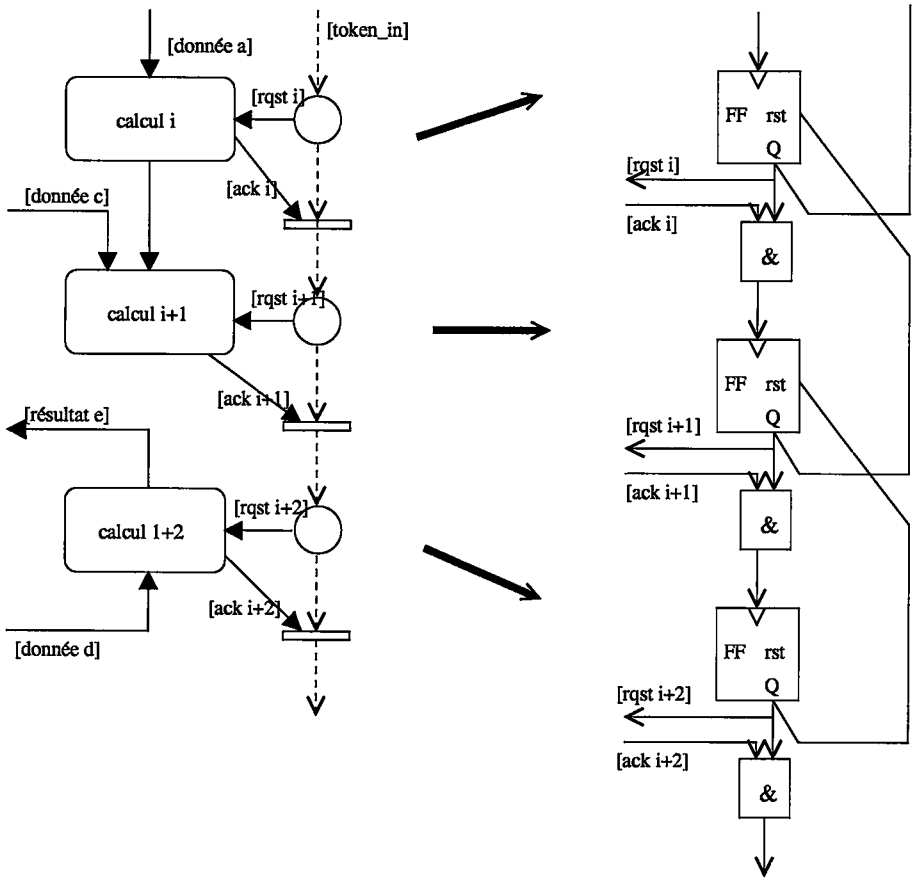

FIG. 4.13 – *Allocation des réseaux de Petri Interprétés à cellules physiques - Bascules*

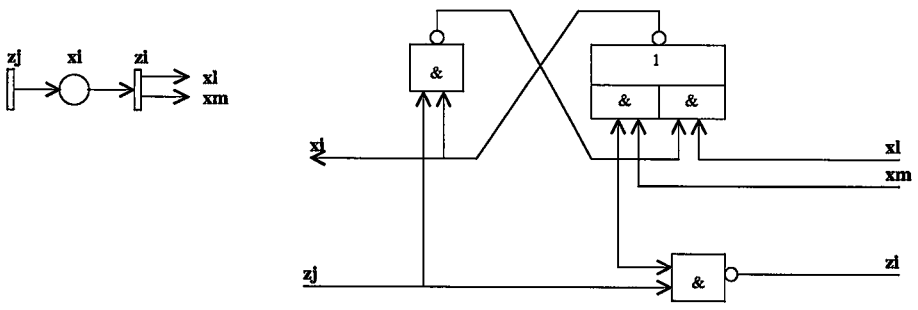

FIG. 4.14 – *Allocation des réseaux de Petri Interprétés à cellules physiques - Portes Logiques*

- cellule d'appel,
- arbitre
- structures génériques des réseaux de Petri ;

4.6 Choix technologiques

L'architecture du D.E.C. a été développée d'une manière indépendante de la technologie de fabrication. Cette méthode est basée sur la réutilisation à tous les niveaux d'abstraction et sur l'utilisation des transformationes correctes pour la génération architecturale. Nous proposons une séparation des différents aspects de la conception pour permettre une meilleure exploration de l'espace de conception. Nous proposons une séparation entre comportement et structure, entre contrôle et transformation de données et entre communication et traitement. Les décisions des choix technologiques arrivent à un moment où les décisions essentielles ont été déjà prises.

Ainsi, le problème de coordination entre le matériel et le logiciel n'est plus un problème principal et il n'apparait pas dans les premieres phases de la conception. Ceci amène des avantages recherchés: une meilleure exploration de l'espace de conception et une meilleure indépendance technologique. Dans le cas du D.E.C., l'allocation d'un bloc structurel à un microprocesseur programmable ou à un processeur numérique du signal n'est pas considérée. L'application ne demande pas la reprogrammation totale et pourtant l'application n'as pas besoin d'un processeur à usage générique. D'autre part, la contrainte de fiabilité et de robustesse aux défaillances oblige à minimiser le nombre de registres et de mémoires de programme.

Du point de vue des communications, l'architecture générée est une architecture orientée interfaces. Cette séparation entre communications et traitement, permet une meilleure exploration du comportement des communications à différents niveaux d'abstraction et une meilleure maîtrise de la complexité. Ainsi, les ressources disponibles au niveau communication seront mieux utilisées et évaluées. Dans le cas du D.E.C., les possibilités technologiques sont variées au niveau de la couche physique et des protocoles de bas niveau (entrée/sortie analogique ou numérique, interfaces optoélectroniques ou microondes, etc). Être independant de ces differents choix permet aux concepteurs d'explorer les possibilités des protocoles de haut niveau plus adaptés et plus simples. Ces architec-

tures ont une tendance vers les systèmes hautement distribués assurant de cette manière un niveau de modularité assez élévé. Ceci demande une intelligence locale pour implémenter des protocoles de haut niveau. Les caracteristiques principales de nos architectures sont, d'une part, la définition de *services* d'interface indépendants des couches de communication de bas niveau et, d'autre part, le développement d'interfaces basées en blocs VHDL-AMS. L'exploration des architectures asynchrones et synchrones est une partie très importante de la conception. Les bonnes propriétés des architectures asynchrones peuvent être mises au profit des applications des Micro Systèmes étant donné que nos architectures sont globalement asynchrones. Ce qui n'interdit pas la possibilité d'utilisation des modules synchrones localement.

En ce qui concerne les interfaces physiques d'un D.E.C., autres que les communications, le choix est vers l'intégration totale du Micro Système ("System on a Chip" [71], MCM, Flip Chip, etc). Néanmoins, les interfaces peuvent être plus importantes en taille que les parties capteurs, actionneurs et intelligence locale et leur géometrie dépend fortement des considérations issues du monde de la microélectronique y compris le packaging et le support physique.

La réalisation physique du microsystème est basée esentiellement en technologie CMOS, demandant des sources de puissance électrique adaptées (5-12 VDC, 1W). La présence de capteurs ou actionneurs du type électrostatiques ou piezoélectriques peut demander l'utilisation de micro sources à haut tension. Notre proposition d'inclure des fonctions de supervision permet des performances spécifiques en autonomie, auto test et fonctionnement dégradé vis a vis de l'alimentation électrique.

D'autre types d'interfaces physiques sont à considérer, telles que la fluidique, la chimique ou la thermique. Ceci introduit des besoins autres qu'électriques: le contrôle des espaces internes du Micro Système pour permettre le passage de la lumière, des flux de matière, etc et l'introduction des connexions différentes aux connexions électriques.

La disponibilité d'une architecture fiable et réalisable permet d'entreprendre des études orientées à la fiabilité des interfaces physiques et des matériaux hétérogènes utilisés pour la fabrication du Micro Système, sans remettre en cause le comportement désiré de l'application.

4.7 Sélection de la meilleure architecture

Des études de faisabilité en micro et nano technologies [10] pour le D.E.C ont permis choisir la technologie d'assemblage du type MCM ("Multi Chip Modules") en cherchant le coût de fabrication le plus bas possible.

Le choix de la technologie ASIC pour réaliser les parties numériques a été guidé par l'utilisation des standards de la microélectronique ce qui apporte des avantages au niveau du prix.

Aucune partie du D.E.C n'a été réalisée en logiciel, étant donné l'impossibilité de réparation, la difficulté de communiquer en opération et les conditions environementales extrêmes.

Les communications physiques se font dans la bande des radio fréquences et à courtes distances (300 m), étant donné que la communication avec le D.E.C se fait seulement quand l'avion est au sol.

Le choix de l'alimentation de puissance électrique permet une autonomie de plusieurs mois d'opération avant la recharge. Cette performance est due aux fonctions d'autonomie du superviseur integré au D.E.C. La recharge des micro batteries peut se faire par une interface microondes à haute puissance.

Nous pouvons constater que le critère de choix a été principalement le coût de la fabrication et de l'intégration technologique. Ce constat part du fait que l'architecture a été validée et vérifiée formellement avant de faire le moindre choix technologique. L'architecture est en elle même fiable et robuste vis à vis des exigences et ceci a été démontré bien avant les choix technologiques. L'analyse des problèmes propres à la fabrication (physique des défaillances, simulations mécaniques et thermiques) peut être menée dans un cadre indépendant du modèle comportemental et structurel.

4.8 Réalisation d'un prototype physique

Dans le paragraphe précédent, nous avons spécifié l'architecture d'un microsystème dont la fonction principale est d'observer l'environnement physique et les contraintes extrêmes auxquelles sont soumis des systèmes mécaniques complexes. Afin de valider ces spécifications et de pouvoir effectuer des opérations de tests, nous allons concevoir une maquette à partir de composants discrets comportant l'ensemble des circuits (capteurs, antenne, circuits électroniques, alimentation,...) qui devront être assemblés selon les techniques dites MCM

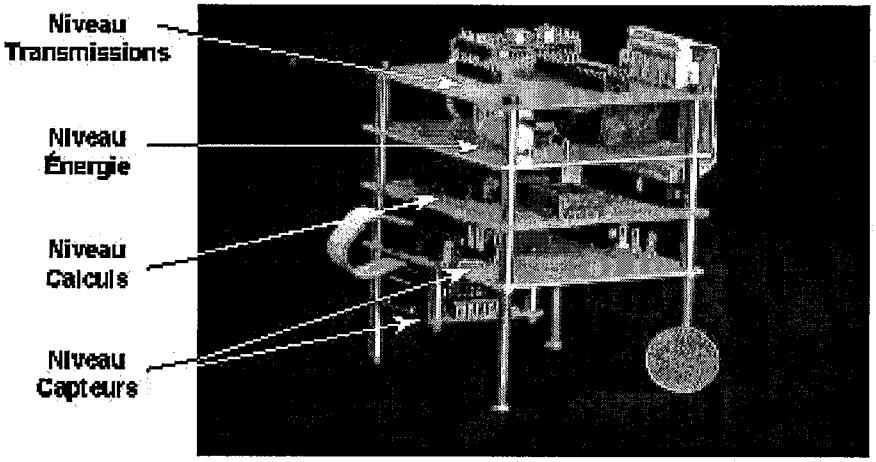

FIG. 4.15 – *Maquette du D.E.C*

(Multi Chip Modules - 3 dimensions). Bien que les dimensions de cette maquette ne soient pas représentatives du microsystème final, notre stratégie est tout de même de disposer les circuits selon leurs natures en plusieurs couches et en insérant des connecteurs verticaux qui assureront les liaisons électriques entre les différents niveaux. Les figures 4.15 et 4.16 illustrent l'aspect global de cette maquette.

4.8.1 Bloc Polariser

Les circuits composant la maquette doivent être alimentés sous une tension de 5V constante avec une capacité suffisamment importante afin d'alimenter correctement le dispositif. Ayant des difficultés à trouver une alimentation répondant à ces critères, nous avons décidé de choisir un pile de 9V délivrant 1,2 A.h.. Pour abaisser cette tension de 9V à la tension nominale, deux options sont possibles: un régulateur à transistor ballast ou un montage abaisseur à découpage.

Pour des problèmes de rendement et de dissipation de puissance trop importante pour le régulateur, notre choix s'est donc porté sur le montage abaisseur de type Buck qui sera réalisé à partir du circuit de référence LM2574N-5.0. Il s'agit d'un modulateur de largeur d'impulsions qui ne nécessite que deux condensateurs, une diode et une inductance comme composant externes, la résistance de

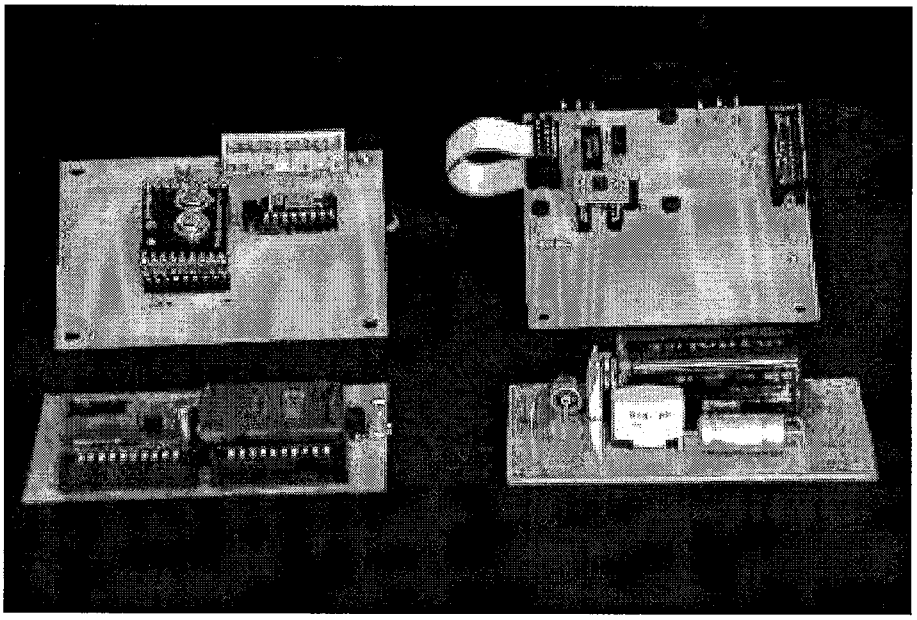

FIG. 4.16 – *Maquette du D.E.C. Desensamblée*

réglage de la tension de sortie est intégrée au circuit permettant d'obtenir une tension de 5V pratiquement constante.

Les micro-relais, dont le rôle est de commuter l'alimentation de certains modules, ont été réalisés à l'aide de transistors MOSFET. La rapidité de commutation, n'étant pas un paramètre critique, nous avons sélectionné les transistors NDP6030PL proposé par la Société Fairchild Semiconductor. Il s'agit de transistors MOS canal P ayant un résistance à l'état ON de 0.6 Ohms, soit une chute de tension de 6 mV pour un courant de 10mA.

4.8.2 Bloc Mesurer

Le Capteur de Température: LM35A - National Instruments Plusieurs types de capteurs de température sont disponibles sur le marché. Notre choix s'est tout d'abord porté sur des capteurs intégrés pour des raisons évidentes d'encombrement. Parmi ceux-ci, il est possible de distinguer deux grandes familles: les capteurs délivrant un courant ou les capteurs délivrant une tension Le principal inconvénient concernant les capteurs à source de courant est qu'ils nécessitent un circuit de conversion courant/tension afin d'être compatible avec le circuit de traitement analogique des signaux. Sachant que nous devons faire

face à des contraintes de consommation électrique et de surface, nous avons préféré opter pour la deuxième grande famille de capteurs, à savoir les capteurs à source de tension. Des capteurs appartenant à cette famille sont proposés par la société National Semiconductors et les principales caractéristiques sont les suivantes: ce capteur fonctionne dans un domaine de température compris entre 0C et +100C. Il délivre une tension de sortie de 10mV/C avec une précision de 0.25 C. L'étalonnage est direct et en C et doit être couplé à une résistance et une capacité.

Les accéléromètres: XMMAS40G - MOTOROLA MOTOROLA propose différents types de capteurs pour des applications principalement dans l'industrie automobile (airbag,...). Ils développent également des accéléromètres trois axes 40G mais qui sont encore au stade du prototypage. Les circuits sélectionnés pour la conception d'un premier prototype sont des accéléromètres mono axe 1G. Le principe est donc de disposer chaque accéléromètre selon un axe particulier (horizontal, vertical ou transversal). Ces circuits sont réalisés à partir de techniques de micro-usinage et intègrent des composants électroniques (amplificateurs, filtres passe-bas, circuits de compensation en température, etc...) qui permettent de fournir en sortie un signal électrique propre et pré-traité. Ils intègrent aussi un circuit d'auto-test permettant de vérifier sur site l'exactitude des mesures prélevées. Dans le cadre d'une coopération [10], la Société MOTOROLA a fournit une "carte fille" comprenant les accéléromètres disposés selon les trois axes avec leur circuit de polarisation et fournissant sur trois broches les signaux électriques proportionnels à l'accélération.

4.8.3 Bloc Conditionner

Les mesures prélevées par les capteurs sont ensuite transmises à une chaîne d'acquisition de données analogiques. Cette chaîne comprend: un échantillonneur / bloqueur, un multiplexeur analogique et un convertisseur analogique / numérique. Dans le commerce, il existe de nombreux composants pouvant assurés les fonctions précédentes, mais nous avons choisi le circuit proposé par Texas Instrument dont la référence est: TLC 540I. Il intègre un multiplexeur analogique à 12 voies, un échantillonneur / bloqueur, un convertisseur analogique / digital à approximations successives qui convertit sur 8 bits toute tension comprise entre Vref+ et Vref- ainsi qu'une interface d'entrée sortie série. De plus,

mis à part son faible encombrement, il est réalisé en technologie CMOS, ce qui lui assure une très faible consommation. Ce circuit fournit donc les mesures de l'accélération selon les trois axes, la valeur de la température et le niveau d'alimentation sous la forme de données numériques, le tout transitant sur une seule voie.

4.8.4 Bloc Mémoriser

Une mémoire du type Flash PROM M29FO40 permettra de conserver les mesures prélevées. Une série de mesure (accélération selon les trois axes et la température) nécessitera une taille mémoire correspondant à quatre octets. Le circuit que nous avons sélectionné a une taille de 4 Mbits, nous pourrons dès lors mémoriser environ 100 000 mesures à l'état brut, valeur que nous pourrons optimiser jusqu'à environ 130 000 mesures si nous utilisons un algorithme classique de compression de données.

4.8.5 Bloc Superviseur

Le coeur de la maquette est constitué d'un FPGA Xilinx XC4010E . Ce circuit est programmé pour effectuer toutes les opérations de traitement des mesures (bloc traiter) et de contrôle de la maquette. Il doit contrôler le séquencement des prises de mesures, autoriser l'alimentation de certaines fonctions du système, contrôler l'enregistrement et la lecture des mesures dans la mémoire, gérer l'émission de mesures ou de signaux d'alerte, imposer le mode de fonctionnement (permanent, veille, dégradé) du dispositif... Le mode de communication asynchrone permet d'optimiser la consommation énergétique du dispositif. En effet, les modules ne seront alimentés que lorsqu'ils seront sollicités. En conséquence, nous avons sélectionné un circuit FPGA autorisant ce type de communication plutôt qu'un microprocesseur du type 68HC11 distribué par MOTOROLA.

4.8.6 Bloc Communiquer

Les communications avec l'unité de contrôle sont réalisées par l'intermédiaire d'une liaison radiofréquence. L'émetteur / récepteur (FMBiM-418F FARNELL) que nous utiliserons est de type half duplex, c'est à dire, qu'à un instant donné, les données qui transitent sur une seule voie ne peuvent qu'être émises ou reçues. Cet émetteur / récepteur fonctionne en modulation de fréquence, avec

une porteuse à 418 MHz. Il est associé à circuit d'interface permettant de fournir des informations en série à partir de données provenant de la mémoire par l'intermédiaire d'une liaison parallèle.

L'émetteur / récepteur utilisé ici émet et reçoit des données en série. Pour assurer une bonne compatibilité entre le calculateur et la mémoire d'une part et ce circuit d'autre part, il faut insérer une interface parallèle (8bits) / série. Pour assurer cette fonction, nous avons choisi le circuit SN74HC165 de Texas Instruments réalisé en technologie CMOS.

4.9 Conclusions

Le cas d'étude présenté dans ce chapitre, nous a permis de développer une famille de produits, associant à la demande, des fonctionnalités diverses, conçues, simulées et fabriquées avec des outils identiques. Cette démarche est attendue comme systématique et elle a pris en compte tous les éléments, tant techniques qu'économiques. La phase de définition des éléments nécessaires à l'élaboration des familles de produits a fait apparaître les étapes technologiques nécessitant des travaux supplémentaires dans les domaines des Micro Systèmes, des capteurs, des outils de conception, des outils de simulation, etc. Ensuite, il est présenté aussi une démonstration de la faisabilité des concepts retenus par le biais de la réalisation de prototypes répondant à des besoins différents.

Les études des technologies disponibles (MEMS, capteurs, interfaçages), les études des logiciels disponibles, la réalisation de prototypes, la validation des logiciels et les études de la qualité, de la fiabilité et de l'intégrité des sous ensembles ont donné lieu à:

- des prototypes de Dispositifs d'Enregistrement de Contraintes (D.E.C) miniaturisés, valides et de fiabilité connue et
- des prototypes de capteurs, Micro Système et logiciels prêts à être intégrés dans les D.E.C.

Les principaux points forts du système multi capteurs étudié sont les suivants:

- son autonomie (Surveillance, Gestion et Changement de modes, Organe d'adaptation à l'évolution de son environnement),

- son électronique basse tension et faible consommation,
- une communication interne asynchrone,
- sa robustesse lui permettant un fonctionnement en mode dégradé, son faible encombrement.

Ce type de dispositif va permettre d'analyser les changements des conditions d'environnement dans lesquelles évoluent les systèmes surveillés avec le temps afin d'améliorer :

- leur fiabilité et par conséquent leur durabilité, en détectant suffisamment tôt la détérioration et les pannes naissantes dans des mécanismes critiques,
- la mémorisation des pannes et des événements exceptionnels afin d'établir un historique,
- la prise des mesures de corrections appropriées.

Chapitre 5

CONCLUSIONS ET PERSPECTIVES FUTURES

5.1 Introduction

Les micro technologies, grâce notamment à l'utilisation des technologies de micro usinage photo lithographique et d'intégration mises au point dans le cadre de la microélectronique, rendent possible la création de produits ultra miniaturisés dont les dimensions vont du millimètre au micromètre. C'est ainsi que sont apparus à la fin des années 80, les micro systèmes qui voient leurs performances augmenter régulièrement, et leurs dimensions, leurs poids, leurs coûts diminuer notablement grâce à l'emploi de procédés de fabrication collectifs déjà exploités en microélectronique. Les micro systèmes sont des systèmes miniaturisés interfaçant le monde non électrique au moyen de capteurs et d'actionneurs, avec un ensemble électronique dont les fonctions principales sont le traitement des informations prélevées et la communication avec l'extérieur. La conception d'un micro système qu'il soit monolithique ou hétérogène pose des problèmes analogues à la conception des systèmes assemblés de manière classique avec les contraintes complémentaires très difficiles liées à l'impossibilité d'apporter des corrections locales sur des produits finis. C'est ici le règne du " sans faute "qui impose une très grande rigueur de conception. A ce titre, la conception des mi-

cro systèmes et la définition de méthodologies performantes est encore un objet de recherche que nous avons tenté d'aborder dans ce travail.

5.2 Les enjeux

Plus généralement, les micro technologies et les micro systèmes constituent un facteur majeur de compétitivité pour les systèmes implantables, portables et embarqués. L'intégration poussée des fonctions s'accompagne d'une économie d'énergie. Les micro systèmes consomment peu et deviennent de plus en plus autonomes. L'autonomie des nouveaux produits doit être comprise comme une capacité propre du micro système à décider mais elle suscite aussi un besoin de communication sans fil sans précédent, besoin que l'association des technologies du numérique et des micro technologies auront à relever dans les années à venir. Les applications vont de la carte à puce sans contact située à quelques millimètres de son lecteur à la transmission d'un ordinateur portable vers un satellite. Malgré les avantages, deux contraintes majeures au développement subsistent encore: La première contrainte concerne l'absence de normes relatives à l'interfaçage des micro systèmes (mécaniques et communication) avec les systèmes déjà existants. En effet, la plupart des fabricants créent et appliquent leurs propres standards aux produits qu'ils proposent. Il est par conséquent difficile de remplacer des systèmes macrocospiques devenant désuets en raison de leurs caractéristiques (volume, poids, consommation, fréquence de fonctionnement,...) par des microsystèmes beaucoup plus compacts et beaucoup plus performants. De la même manière, il est difficile de remplacer des micro systèmes par d'autres micro systèmes ayant les mêmes fonctionnalités mais proposés par des fabricants différents. Néanmoins, des études menées par des grands centres de recherche et des regroupements d'industriels doivent permettre d'aboutir à des normes internationales concernant l'interfaçage des micro systèmes avec leurs environnements. L'autre contrainte a concerné notre sujet de recherche: les méthodes de conception des micro systèmes. Comme dans les grands systèmes assemblés, différents domaines pluridisciplinaires sont impliqués dans le processus de conception (électronique, mécanique, optique, électromagnétique, télécommunication, biochimique, informatique,...). Le développement des micro systèmes requiert donc l'intervention d'experts et l'utilisation d'outils informatiques très puissants et complémentaires.

5.3 Analyse de notre contribution

Nous avons mis en place une méthodologie de conception, qui permet aux experts de secteurs différents de travailler ensemble avec des outils communicants pour servir l'objectif de réalisation d'un produit répondant à des spécifications précises. Le caractère intégré du produit impose, comme c'est le cas en microélectronique, le " sans faute " : aucune retouche, aucun ajustement, aucune intervention n'est possible sur un produit achevé. Notre démarche pour atteindre cet objectif a été d'analyser les approches décrites dans la littérature ou déjà exploitées et de proposer une approche synthétique globale qui réponde aux exigences déjà évoquées de collaboration pluridisciplinaire et de sûreté de conception. Notre guide est la pratique actuelle de la conception microélectronique d'où nous tirons trois directives :

- La réutilisation des acquis sous la forme d'une insertion de sous-ensembles déjà conçus et validés,
- La modélisation physique et comportementale de tous les constituants du système associé à une simulation globale,
- La démarche descendante, qui sous la condition des deux directives précédentes, permet d'aller des spécifications aux produits, étape par étape.

Cette troisième directive pose la question de la définition des étapes de conception : c'est une question très difficile car il est rare qu'un système ne soit issu de la modification incrémentale des concepts déjà existants, prouvés et validés. Nous considérons ici que le produit est totalement nouveau. Il part donc d'une *idée de produit* qui fixe une première étape d'évaluation projective conduisant à des *spécifications générales.* C'est une étape préalable très riche, très innovante, très liée aux capacités créatrices des promoteurs qui sort de notre réflexion. Nous sommes donc partis d'un niveau où les spécifications générales ont été établies et nous avons identifié deux étapes supplémentaires qui conduisent à une conception en trois étapes :

- Spécification du produit,
- Conception générique du produit,
- Conception matérielle,

qui appelle les commentaires suivants : Pour la définition d'une conception générique, nous argumentons sur le fait que la tendance est d'aller vers une conception *hors de toutes contraintes technologiques*, nous l'appelons *générique*. Elle traduit les spécifications générales en comportements et structures temporisées à un niveau où on peut innover par exploration de l'espace de conception, on va établir qu'il y a cohérence fonctionnelle, que l'organisation fonctionnelle est viable. Nous avons essayé de montrer l'intérêt de cette étape et la façon que nous voyons de l'enrichir par une base de données de haut niveau d'agrégation. Pour la conception matérielle, nous avons vu surtout la difficulté du travail pluridisciplinaire : notre option était de proposer une simulation globale du système en utilisant le standard VHDL-AMS et en laissant à chaque discipline le soin de gérer son propre travail de conception mais en exigeant que le dialogue puisse s'établir avec la simulation globale que nous appelons " prototype virtuel ", notamment par la disponibilité de modèles physiques et comportementaux. Un point essentiel de cette approche est la transition entre conception générique et conception matérielle, que nous avons proposé de gérer par deux principes :

- la traduction des concepts génériques retenus en langage VHDL-AMS,
- la mise en place d'une base de données cohérente et unique pour toutes les étapes de conception.

Notre démarche a été pragmatique. Elle a visé simplement à identifier les bons comportements de conception et à les mettre en forme dans une démarche globale en validant autant que possible nos choix. Pour ce faire, nous avons choisi de nous appuyer sur une conception exemplaire de micro système spécifique. Nous avons utilisé comme support de démonstration la conception d'un micro système dont le fonctionnement est basé sur la fusion multisensorielle. L'idée est de surveiller l'état vibratoire d'une structure mécanique par des micro systèmes répartis en différents points de la structure observée. Elle se fonde sur la miniaturisation croissante pour former des spécifications générales d'une surveillance répartie en termes de modules autonomes, petits (quelques cm3) et communicants. Ce système est formé donc de plusieurs Dispositif d'Enregistrement des Contraintes (D.E.C) ; un D.E.C comportant cinq capteurs mesurant l'accélération selon les trois axes, la température et l'environnement électromagnétique. Il peut également communiquer de manière bidirectionnelle avec l'extérieur par liaison sans fil. Ce dispositif, disposé sur les endroits stratégiques nécessitant une surveillance particulière, permettra en ne conservant en mémoire que les valeurs

significatives, de retracer l'évolution temporelle des contraintes environnementales qu'ils ont subies. A partir de ces informations, il sera possible d'anticiper le remplacement de certains équipements sensibles si les contraintes mécaniques ont été très importantes, ou au contraire de le retarder si celles-ci ont été très supportables.

Nous avons introduit la notion de Superviseur Intégré qui a, pour accomplir sa tâche, accès à différentes valeurs dans le système environnant, une capacité propre de calcul, la possibilité de générer des tests, des commandes et de conserver en mémoire des données pertinentes, ainsi que la possibilité d'échanger des informations avec l'extérieur. Pour ce genre de dispositifs, il existe de nombreuses et très diverses applications. Notre tâche a été alors de prévoir une adaptation possible de l'architecture du circuit électronique de base, le Superviseur Intégré, aux types de capteurs qui lui fourniront les mesures analogiques ainsi qu'au mode de communication (filaire/sans fil et Optique/Radiofréquence) afin d'assurer la fonctionnalité souhaitée. Pour atteindre cet objectif, nous avons créé une bibliothèque de composants implémentés sur le standard VHDL-AMS. Nous avons atteint dans ce travail de thèse le but recherché:

proposer une méthodologie, une méthode et un modèle pour aider à la conception générique et architecturale des micro systèmes spécifiques ayant comme caractéristique principale un comportement auto temporisé, Ce travail n'aurait pu être conduit à terme sans la contribution de mon collègue Noursaid HARCHANI qui a pris à son compte une problématique très proche, celle de la *conception matérielle*, nous permettant à deux, de couvrir les deux niveaux de conception électronique. Notre profil personnel de concepteur nous a amenés à privilégier une présentation favorisant d'abord les aspects de démonstration pour en tirer des arguments en faveur d'une démarche globale de conception.

La distinction entre conception générique et architecturale et conception matérielle a été effectuée sur les critères de non-référence aux choix technologiques:

- Le niveau générique manipule des fonctions générales et valide les choix architecturaux et temporels,

- Le niveau matériel gère la pluridisciplinarité en faisant collaborer des technologies et des disciplines différentes.

L'idée exposé dans ce travail est que cette collaboration s'opère sur une plateforme commune: le prototype virtuel décrit dans un langage commun VHDL-

AMS. La communication entre la plate-forme commune et chaque discipline se faisant sur fond de modélisation comportementale:

- Le concepteur, sur la base de la conception générique, alloue des fonctions à réaliser avec des spécifications précises et des tolérances. Il transfere sur simulateur VHDL-AMS les modèles comportementaux qui lui sont transmis et construit le *prototype virtuel*
- Le fournisseur fait sa propre conception des sous-ensembles qui le concerne avec ses outils et ses habitudes, mais il en communique le résultat sous une forme compatible en terme de modélisation comportementale: tableau de données, modèles identifiés, etc.
- Concepteurs et fournisseurs revoient si nécessaire les spécifications et les tolérances selon une procédure d'optimisation.

Dans la mesure où modèles et technologies correspondent, la fabrication globale doit conduire à une réalisation finale immédiatement opérationnelle. C'est le bénéfice immédiat, mais il y a aussi à considérer:

- L'effet de capitalisation au niveau générique, capitalisation de savoirs validés et expériences en termes génériques,
- L'effet de capitalisation des modèles partiels qui peuvent être réutilisés dans d'autres projets,
- L'effet de capitalisation du prototype virtuel qui peut être disponible pour traiter des modifications à apporter durant la vie du produit.

Nous avons essayé de donner un modèle de représentation architecturale qui réponde aux besoins de la conception matérielle:

- les difficultés d'interfaçage intra et inter micro systèmes, pour que l'on considère l'élaboration de l'architecture d'un micro système comme un assemblage de blocs en tenant en compte les interfaces pluridisciplinaires entre ces blocs;
- des micro systèmes fiables, autonomes, à basse consommation de puissance électrique et à basse émission de bruit électromagnétique.

L'approche descendante proposée permet une nouvelle conception des Micro Systèmes partant des exigences et non de la microélectronique, tel qu'il est courant aujourd'hui.

La partition fonctionnelle proposée de l'architecture interne d'un Micro Système offre un cadre approprié pour la modularité et pour la conception aval.

HiLeS a amené une conception modulaire qui permet le passage direct du réseau de PETRI représentant le fonctionnement du système désiré à une réalisation physique indépendante de la vitesse.

La description hiérarchique par réseaux de Petri permet une spécification et une validation locales et pas seulement globales, ce qui rend utile la méthode pour le traitement des structures complexes. Vérifier les bonnes propriétés de façon locale ne garantit pas une optimalité stricte, par contre, elle rendra possible le traitement de systèmes complexes très rapidement et sans difficulté.

5.4 Perspectives

Comme perspectives de notre travail, nous pouvons mentionner :

- La validation de la méthodologie en fabriquant des micro systèmes multi capteurs pour des applications connexes (naval, environnement, ponts et chaussées, etc.) ;
- la conception de l'unité de contrôle éloignée
- l'augmentation de l'intelligence locale du D.E.C (gestion de mémoire par associations, mesures de vieillissement, etc.)
- le développement d'une organisation de micro systèmes et pas seulement un groupe coordonné ;
- l'enrichissement de la librairie globale ;
- le développement d'une librairie des éléments micro architecturaux pour compléter la synthèse des ASICS, MCM et SoC en vue d'une réalisation asynchrone.
- l'exploration de l'approche ascendante en vue de la macro modélisation.

Annexe A

La méthode SA-RT (Structured Analysis for Real Time)

L'analyse structurée temps réel a été définie au milieu des années 80 par deux équipes de chercheurs Ward et Mellor (WM) et Hateley et Pirbhay (HP). Leurs travaux, effectués séparément, proposent des extensions temps réel de l'analyse structurée de DeMarco. Ces extensions portent sur l'adjonction des fonctionnalités de contrôle décrivant la dynamique du système et le traitement des données correspondantes (les signaux événementiels).

Cette méthode est basée sur l'élaboration de deux modèles, un modèle logique de spécification d'un système, appelé modèle essentiel (Ward et Mellor) ou modèle des besoins (Hateley et Pirbhai), et un modèle technologique ou de conception désigné par modèle d'implementation (Ward et Mellor) ou modèle d'architecture (Hateley et Pirbhai).

Le modèle technologique, issu de la conception, porte sur l'architecture d'un système. C'est un début de solution informatique, à la fois matérielle et logicielle, liée au choix d'une technologie particulière, et apte à supporter les fonctionnalités, à résoudre les contraintes, et à atteindre les performance spécifiées.

Le modèle logique, issu de la spécification et qui nous intéresse plus particulièrement, concerne les aspects fonctionnels, informationnels et événementiels d'un système temps réel. L'aspect fonctionnel porte sur les fonctionnalités que

le système doit présenter et l'aspect informationnel concerne les données mises en jeu pour leur accomplissement; ces deux aspects sont communs à tous les systèmes informatisés. L'aspect événementiel est spécifique des systèmes temps réel, qui doivent répondre à des signaux en provenance de l'environnement, et réaliser des transformations de données conditionnées par des contrôles.

A.1 Aspect fonctionnel de la méthode SA-RT

L'aspect fonctionnel de la méthode SA-RT est modélisé à l'aide des outils de l'analyse structurée. Les composants graphiques du modèle sont les flots de données, les stockages de données et les processus. Ils sont organisés à l'aide de règles de construction précises sous forme de diagrammes de flots de données (DFD).

Le modèle est organisé sous forme d'une hiérarchie de DFD, dont le diagramme de contexte représente le niveau le plus haut. Ce diagramme est la seule représentation dans laquelle figurent les bords du modèle; ceux-ci délimitent une frontière entre le système et son environnement.

Chaque DFD décompose un processus d'un DFD de niveau supérieur. Le processus décomposé est appelé processus père et le DFD qui le décompose est dit enfant. Le passage d'un processus père à son DFD enfant doit respecter la règle de conservation des interfaces. Un processus élémentaire qui ne peut pas être décomposé sous forme de DFD est un processus primitif. Il constitue les feuilles de l'arborescence et il faut recourir à un autre mode de représentation pour décrire comment il produit ses sorties en fonction de ses entrées. Ce stade ultime de la modélisation porte le nom de spécification de processus et fait appel à un mode de représentation principalement textuel, algorithmique, pseudo-langage,... .

A.2 Aspect événementiel de la méthode SA-RT

L'aspect événementiel d'un système temps réel (STR) nécessite d'être modélisé à l'aide d'outils de représentation appropriés, qui prennent en compte les événements qui conditionnent l'activité des processus et qui permettent de représenter la logique de contrôle du système. Les approches WM et HP de l'analyse structurée temps réel proposent des représentations différentes du contrôle d'un STR.

L'approche WM, intègre l'aspect événementiel dans la représentation de l'aspect fonctionnel en décrivant les flots de données, les flots de contrôle, les processus (appelés transformation de données) et les unités de contrôle (nommées transformation de contrôle) sur les mêmes diagrammes appelés schémas de transformation. Les relations d'activation/déactivation y figurent de manière explicite. Ce choix de représentation offre l'avantage de donner une vue d'ensemble de l'activité contrôlée d'un STR, mais nécessite des règles de formation spécifique pour distinguer les traitements du contrôle. Cette distinction est obtenue en utilisant les lignes continues pour l'aspect fonctionnel et des lignes pointillés pour l'aspect événementiel (contrôle).

Les événements sont du type interruption. La logique de contrôle est hiérarchisée de la même façon que les DFD et peut être spécifiée à l'aide d'automates à états finis, de tables ou de matrices état transition.

L'approche Hatelay-Pirbhai de l'analyse structurée temps réel propose d'intégrer l'aspect événementiel (contrôle) d'un STR dans son aspect fonctionnel, en représentant séparément mais de façon couplée, d'un côté, les flots de données et les processus, à l'aide de DFD, et de l'autre, les flots de contrôles correspondants, à l'aide de diagrammes comparables aux DFD, appelés Diagramme de Flots de Contrôle (DFC). Les DFC sont hiérarchisés comme les DFD et chaque DFC va de paire avec un DFD de niveau correspondant.

Un DFC contient des processus (les mêmes que ceux du DFD qui lui est associé), des flots de contrôles et des barres de spécification de contrôle. Les barres de spécification de contrôle marquent l'interface entre un DFC et la spécification de sa logique de contrôle.

Les activeurs, ou contrôles de processus ne sont pas représentés dans un DFC, ils n'apparaissent que dans la spécification de sa logique contrôle.

Les contrôle sont de type changement de valeur de données continues, principalement binaires.

La spécification de la logique de contrôle, notée CSPEC dans la terminologie HP, est obtenue à l'aide de diagrammes, de tables ou de matrices état-transition complétés par une table d'activation des processus qui résume les différentes activations : un 0 indique qu'un processus n'est pas activé et un nombre strictement positif indique son ordre d'activation ; si un tel ordre n'est pas important, un 1 suffit.

Ce choix de représentation offre l'avantage de séparer les rôles distinctifs des traitements et du contrôle sans faire appel à trop de règles de formation

spécifiques en même temps qu'il réduit la quantité d'information contenue dans un diagramme. Il nécessite cependant de mener en parallèle les vues des deux types de diagrammes pour avoir une perception globale de l'activité contrôlée d'un STR.

A.3 Aspect informationnel de la méthode SA-RT

Les données et les événements qui interviennent à tous les stades du modèle sont définis dans un dictionnaire de données. Quant aux stockages des données complexes, l'approche WM prévoit une modélisation à l'aide de diagramme Entité/Association servant à modéliser des données de taille importante et d'organisation complexe, comme par exemple des bases de données.

Enfin, l'approche WM préconise la spécification des exigences temporelles qui concernent uniquement les interfaces avec l'environnement. Les taux de transferts des données externes sont spécifiés dans le dictionnaire des données, alors que le temps de réponse aux événements externes sont décrits dans des tables de spécification éventuellement complétées par des chronogrammes.

Ainsi, la méthode SA-RT fournit un ensemble de techniques intéressantes pour mener à bien la spécification des STR. C'est une approche structurée qui donne le moyen au concepteur de mener une spécification modulaire et hiérarchisée de façon à maîtriser la complexité des systèmes à concevoir. Cette approche propose des outils qui couvrent les différents aspects des systèmes à concevoir. Cette approche propose des outils qui couvrent les différents aspects des STR : fonctionnels, événementiels et informationnels. Cependant, nous pouvons constater deux limitations essentielles de cette méthode pour le domaine d'application qui nous concerne.

Une première limitation concerne les outils de spécification de la logique de contrôle. Les formalisme de représentation des automates, le plus souvent utilisés, sont inappropriés à la description de la coordination et du pilotage dans les Micro Systèmes. En effet, les machines à états finies s'adaptent mal à la spécification des systèmes complexes multitâches et grand degré de parallélisme et cela à cause de l'explosion rapide du nombre des états ;

La deuxième limitation concerne la hiérarchisation des signaux nécessaires à la gestion des modes de fonctionnement. Les méthodes SA-RT n'offrent aucun

formalisme graphique qui permet de représenter et de gérer cette hiérarchie. Les relations exprimées sont de type horizontal et le modèle obtenu par la spécification est plutôt "plat".

Annexe B

Réseaux de Petri Interprétés

Un réseau de Petri est un graphe fini qui comprend deux types de sommets : des places et des transitions. Un arc ne peut relier que deux sommets de type différent. Ainsi, l'ensemble de places se rapprochent de la notion d'état, l'ensemble de transitions et d'arcs décrivent la manière dont le réseau peut évoluer et la distribution des jetons dans les places donne l'état du système.

Un marquage est une répartition de jetons dans les places du réseau. Le nombre maximum de jetons contenus dans une place est egale à un pour les réseaux binaires.

Une transition est dite sensibilisé par un marquage donné si toutes les places d'entrée de la transition sont marquées et que l'interprétation associée à la transition est vraie.

Le franchissement d'une transition est instantané et comporte trois étapes indissociables :

Ces règles d'évolution mettent en évidence des relations de cause à effet du système à événements discrets modélisé. Les places d'entrée et l'interprétation de la transition sont les conditions de l'évolution modélisée par la transition, tandis que les actions et les places de sortie en montrent explicitement les conséquences.

Parce qu'ils permettent une modélisation facile du parallélisme (plusieurs jetons dans le même réseau), de la synchronisation (transition avec plusieurs arcs d'entrée) et de l'alternative (place avec plus d'une transition de sortie), ils sont très utilisés dans le domaine des systèmes à événements discrets.

Une autre facette des réseaux de Petri est la possibilité de faire la preuve formelle de certaines propriétés dites "bonnes propriétés".

Un réseau de Petri est dit vivant pour un marquage donné si, à partir d'un marquage accesible quelconque, toute transition peut être sensibilisée après une séquence de tir appropriée. Pour le système physique modélisé, cette propriété du modèle traduit la non existence de situation de blocage.

Quelle que soit l'évolution du réseau de Petri borné, il existe une limite finie au nombre de jetons dans le réseau. La correspondance avec un système réel, donc la capacité est toujours physiquement limitée, est là aussi immédiate.

Dans une réseau réinitialisable, il est toujours possible de trouver une séquence de tir qui ramène au marquage initial. Cette propriété est très utile lorsque on modélise (et c'est très souvent le cas) des systèmes à caractère cyclique ou répétitif.

Des méthodes d'analyse par réduction du graphe, arbre des marquages accessibles ou recherche d'invariants permettent l'analyse des réseaux de Petri et autorisent donc une forme de validation des spécifications établies à l'aide de cet outil.

Pour décrire un ensemble d'automatismes logiques communicants, on peut associer des réceptivités (fonctions logiques des variables du système y compris les entrées) aux transitions et les commandes ou ordres d'éxecution d'une tache sont associées aux places. Quand une place possède un jeton, la tâche que lui est associée est activée et elle est mantenue jusqu'à ce que le jeton soit enlevé.

Lorsqu'un système "temps réel" est décomposé en une partie commande et une partie données, les opérations effectuées sur les données correspondent bien à des changements d'état et pourtant elles sont associées aux transitions.

Le réseau de Petri ne représente que la partie commande et il est utile de compléter ce dernier par un graphe de données. Un tel graphe est la représentation physisque du cheminement des données. Il définit les données et explicite la menière selon laquelle ces données sont utilisées et stockées. Le système complet sera donc spécifié par un graphe de données et un graphe de commande. A ces deux graphes , on ajoute une "interprétation" pour expliciter les symboles associés aux transitions.

B.1 Le graphe de données

Les noeuds du graphe sont des blocs fonctionnels. Un canal continu lie deux bloc fonctionnels si les données fournies par un bloc sont utilisées par l'autre bloc. Un canal continu lie une place à un bloc fonctionnel, si la présence de jeton dans cette place doit activer la tache du bloc fonctionnel et un canal continu lie un bloc fonctionnel à une transition, si une condition logique doit être associée comme receptivité de cette transition.

B.2 Le graphe de commande

Ce graphe est un réseau de Petri interprété. A chaque transtion du réseau est associé un canal continu provenant d'un bloc fonctionnel du graphe de données. Ce canal associe à la transition k une fonction booléenne Qk definie par le bloc fonctionnel correspondant. Les places en aval de la transtion représentent la liste des opérations Ok qui doivent s'exécuter lors du tir de la transition k. Chaque place aura un canal de sortie, chacun faissant appel au bloc fonctionnel correspondant à l'opération demandée. Si la condition logique Qk est "vrai" et si la transition k est sensibilisée par le marquage du réseau, alors k peut être tirée. Les opérations associées aux places de sortie de k seront activées chaque fois que la transition k sera tirée.

Annexe C

"Specification and Description Language" (SDL)

SDL est un langage de description et de spécification normalisé par ITU (Union de Télécommunication Internationale). Le langage avait évolué depuis la première recommandation en 1980, 1984, 1988 et 1992 où des dispositifs orientés objet ont été inclus dans le langage. SDL est largement répandu dans le domaine de télécommunications. SDL n'est pas dirigé spécifiquement à décrire des services de télécommunications, mais c'est un langage de description tout usage pour des systèmes de transmission. La base pour la description du comportement sont les machines à état étendues communicantes, qui sont représentées par des processus. La communication est représentée par des signaux et elle peut avoir lieu entre les processus ou entre les processus et l'environnement du modèle du système. Quelques aspects de communication entre les processus sont étroitement liés à la description de la structure de système. Une machine étendue d'état se compose d'un certain nombre d'états et d'un certain nombre de transitions reliant les états. Un des états est indiqué comme initial.

Les domaines d'application de SDL sont les systèmes temps réel, interactifs et distribués. Le type d'information est structurel et comportemental à des niveaux d'abstraction du géneral au détaillé.

C.1 Comportement d'un système

Le comportement d'un système est décrit par un nombre de processus dans le système. Un processus est une machine à états étendue, fonctionnant en concurrence avec d'autres processus et de manière autonome. La coopération entre les processus est de type asynchrone par le passage de messages discrets (signaux). Un processus peut recevoir et envoyer des messages de et à l'environement du système. Celui ci doit obéir aux contraintes données par la description SDL du système. Le comportement du système est déterministe et réactif. Chaque processus a une identité unique. Un signal porte l'adresse de l'émetteur et du recepteur du message, en plus des données de travail. Le processus recepteur connait toujours l'adresse de l'émetteur du message. Un processus a une mémoire pour stocker ses variables et l'information de son état. Un processus ne peut pas écrire sur les variables d'autre processus.

Un processus a une file d'attente d'entrée de messages de longueur infinie. Un processus est en état d'attente ou il est en train de faire un transition entre deux de ses états. Une transition est initiée par le premier message de la file. Le message est enlevé de la file d'attente. Pendant la durée d'une transition, les données peuvent être manipulées, des décisions peuvent être prises, des nouveaux processus peuvent être crées et des messages peuvent être envoyés.

Annexe D

VHDL-AMS

IEEE VHDL 1076-1993 (le langage de description de matériel VHSIC) est conçu pour la description et la simulation des systèmes digitaux. La norme approuvée la première fois en 1987 et mise à jour en 1993, a gagné une grande acceptation les dernières années. Des extensions à VHDL-1076 ont ajouté la capacité de décrire et de simuler des systèmes continus mélangés avec des systèmes conservatifs, non-conservatifs, continus et discrets. Le travail a été exécuté sous les auspices du Comité de Normalisation de la Conception Assistée par Ordinateur de l'IEEE et le Groupe de Standardisation de VHDL, sous PAR 1076.1. Le langage étendu s'appelle officieusement VHDL-AMS.

D.1 Les éléments du IEEE VHDL 1076.1

Les aspects continus du comportement des systèmes à paramètres concentrés peuvent être décrits par des systèmes des équations algébriques et différentielles ordinaires (DAEs) avec le temps comme variable indépendante de la forme F(x, dx/dt, t) = 0 où F est un vecteur des expressions, x est un vecteur des inconnues, dx/dt est un vecteur des dérivés des inconnues, et t est le temps.

La plupart des équations de tels systèmes n'ont aucune solution analytique, dans la pratique les solutions doivent être approximées en utilisant des techniques numériques. Les DAEs ont été étudiées intensivement par les mathématiciens numériques au cours des 20 dernières années. Les DAEs peuvent être classifiées selon leur structure (l'indice structural), et plusieurs méthodes de solutions existent pour DAEs d'indice faible. Les algorithmes échouent pour les

systèmes d'indices élevés, mais des méthodes ont été développées qui permettent à de tels systèmes d'être augmentés tout en réduisant l'indice structural. Bien qu'il y ait des issues ouvertes, la théorie des DAEs est suffisamment mûre pour compter pour la solution numérique du DAEs. VHDL 1076.1 fournit une notation pour les DAEs, mais il ne fournit pas une technique pour leur solution. Ainsi VHDL 1076,1 peut rester neutre en ce qui concerne la sélection des méthodes numériques. Il est suffisant que la définition de VHDL 1076.1 décrive le système des équations. L'algorithme de solution est désigné sous le nom de "résolveur analogique". Seulement les résultats que résolveur analogique fournit, et non son algorithme, sont caractérisés dans la définition du langage.

Les inconnues dans la collection de DAEs implicite par le texte d'un modèle sont des fonctions analytiques de temps; c'est-à-dire, ils sont continus à morceaux avec un nombre fini de discontinuités. Le résolveur analogique résout pour les valeurs de tous les inconnues avec le temps, d'abord en convertissant, en utilisant des méthodes appropriées de discrétisation, et puis résolvant les équations aux differences algebriques simultanément.

Ni les variables, ni les signaux obtiennent leurs valeurs de cette manière. Les variables de VHDL obtiennent leurs valeurs par affectation séquentielle. Les valeurs des signaux de VHDL sont calculées des valeurs de leurs sources. Pour cette raison, VHDL 1076,1 présente une nouvelle classe des objets, la quantité, pour représenter les inconnues dans les DAEs. Les valeurs de toutes les quantités sont calculées simultanément par le résolveur analogique. Les quantités doivent avoir des sous élements scalaires du type virgule flottante, pour approximer les nombres réels. On ne permet pas des éléments de type discret parce que les fonctions discrètes du temps ne sont pas des fonctions analytiques. Un objet quantité peut apparaître dans une expression ou la valeur d'une quantité est permise n'importe où. Les caractéristiques d'une quantité composée sont simplement l'agrégation des caractéristiques de ses sous élements scalaires. Le comportement de chaque sous élement scalaire est indépendant des autres.

Des quantités peuvent être déclarées n'importe où un signal peut être déclaré, excepté à l'intérieur d'un module. L'instruction suivante déclare deux quantités de q1 et q2 de type réel: QUANTITY q1, q2: real;

Une quantité peut également être déclarée comme élément d'interface dans un port. Un élément d'interface de quantité s'appelle un port de quantité, par analogie avec des ports de signal. Les quantités d'interface supportent la modélisation par flot de signaux. Par exemple, voici la déclaration d'entité par flot

de signal d'un additionneur de deux entrées avec deux quantités d'interface de mode IN et d'une quantité d'interface de mode OUT:

```
ENTITY adder IS
PORT (QUANTITY in1, in2: IN real; QUANTITY sum: OUT real);
END ENTITY adder;
```

En plus des quantités explicites, les quantités suivantes sont implicitement définies. La dérivée du temps de toute quantité Q: Q'Dot, l'intégrale d'une quantité Q, du temps zéro au temps actuel: Q'Integ, la valeur d'une quantité Q à un intervalle fixe dans le passé (retard idéal): Q'Delayed(t), une approximation à valeurs réelles au temps actuel de simulation: ANOW.

D.2 Les Systèmes Conservatifs

Les systèmes à sémantique conservative – par exemple, les systèmes électriques obéissant aux lois de Kirchhoff – méritent un traitement séparé parce qu'ils sont très utilisés. Une syntaxe et la sémantique spéciales peuvent fournir une notation simplifiée qui réduit le risque d'erreurs et améliore de ce fait la productivité. Dans VHDL 1076,1, les équations décrivant les aspects conservatifs d'un tel système n'ont pas besoin d'être explicités. Seuls les équations constitutives demeurent la responsabilité du concepteur. Dans VHDL 1076,1 la description des systèmes conservatifs utilise un modèle conceptuel graphique basé sur des graphes. Prenons par exemple, un réseau électrique. Ici, les sommets du graphe représentent des noeuds équipotentiels dans le circuit, et les arêtes représentent les branches du circuit par lesquelles le courant passe. Ce modèle conceptuel ne dicte aucune contrainte particulière pour le résolveur analogique. Les quantités de branche représentent les inconnues dans les équations décrivant les systèmes conservatifs. Il y a deux genres de quantités de branche: des quantités "au travers" et des quantités "traversante". Les quantités "au travers" représentent des grandeurs tels que la tension, la température, ou la pression. Les quantités "traversantes" représentent des flots tels que le courant, le flux de chaleur, ou le débit d'un liquide. Les équations constitutives des systèmes conservatifs sont habituellement exprimées en associant une quantité "au travers" et une quantité "traversante" à une ou plusieurs branches. Par exemple, une résistance est modélisée par une branche simple, et son équation constitutive (la loi de l'ohm) associe la tension à travers (quantité au travers) et le

courant sur la résistance (la quantité traversante). Une quantité de branche est déclarée par rapport à deux terminaux. Le terminal est le deuxième nouvel objet de VHDL 1076.1.

Un terminal est déclaré comme étant d'une certaine nature simple, ou d'une nature composée des sous élements scalaires qui sont de nature simple. Chaque nature simple représente une discipline physique distincte – électrique, thermique, hydraulique, et ainsi de suite. A titre d'exemple, les instructions suivantes déclarent deux sous types, la tension et le courant; une nature simple, l'électrique; deux terminaux de cette nature; et une quantité au travers et deux quantités traversantes entre les deux terminaux. La quantité "au travers" représente la différence de potentiel entre les deux terminaux, et les quantités traversantes représentent deux branches parallèles.

```
SUBTYPE voltage IS real;
SUBTYPE current IS real;
NATURE electrical IS
voltage ACROSS;
current THROUGH;
TERMINAL t1, t2: electrical;
QUANTITY v ACROSS i1, i2 THROUGH t1 TO t2;
```

Un terminal peut être déclaré n'importe où. En particulier, un terminal peut être un élément d'interface dans une liste de déclaration de port. Dans ce cas, s'appelle un port terminal. Par exemple, le code suivant déclare les ports terminaux d'une diode:

```
PORT (TERMINAL anode, cathode: electrical);
```

L'association des ports terminaux est employée pour construire des descriptions hiérarchiques d'une manière analogue aux ports de signal utilisés dans les descriptions hiérarchiques numériques.

D.3 Commandes simultanés

VHDL 1076,1 complète les commandes séquentielles et simultanées de VHDL-1076 avec une nouvelle classe "simult" pour la notation des équations différentielles et algébriques. La forme de base est la commande simultanée simple, qui

a la syntaxe suivante:

```
[label:] expression == expression ;
```

Par exemple, l'équation d'une resistance, peut s'ecrire:

```
i == v / r;
```

où I est le courant traversant la résistance, v est la tension au travers une résistance, et r est une constante représentant la valeur de résistance. Les expressions peuvent avoir des valeurs composées, dans ce cas il doit y avoir un sous élement du côté gauche pour chaque sous élement du côté droit. Les expressions peuvent inclure des signaux, des quantités, des constantes, des variables et des fonctions pures. Des appels aux fonctions impures sont rejetés. Quand le résolveur analogique a correctement établi les valeurs de chaque quantité, les sous élements des expressions seront (approximativement) égaux.

L'exemple suivant d'un commutateur simple ilustre les nouvelles caracteristiques du langage:

```
ENTITY ideal_switch IS
GENERIC (closed: boolean := true);
PORT (TERMINAL t1, t2:electrical);  -- terminal ports
END ENTITY ideal_switch;

ARCHITECTURE one OF ideal_switch IS
QUANTITY v ACROSS i THROUGH t1 TO t2;  -- branch quantities
BEGIN IF closed USE  -- simultaneous if statement
v == 0.0;  -- simple simultaneous statement
ELSE i == 0.0;
END USE;
END ARCHITECTURE one;
```

Annexe E

Prototype Virtuel du D.E.C.

E.1 Tables d'entrées et de Sorties

E.1.1 Bloc Traiter

0.1.3	TRAITER	
INPUTS	OUTPUTS	OBSERVACIONES
STRT-TM		Empezar a tomar medidas (viene del bloque supervisor[0.1.8])
PRIM-DIAG[4..0]		Diagnostico de los sensores (viene del bloque tests[0.1.9])
COMMD-SENS[4..0]		Sensores a tomar medidas (viene de la Unidad de Control)
FUNC-MODE[1..0]		modo de funcionamiento (viene del supervisor[0.1.8])
Write-cntr		Senal que activa la escritura en la memoria de umbrales de memorizacion (viene de la U. de control)
sensorin [7..0] (MeasureIN)		Medida del sensor. Viene del bloque Conditionner[0.1.2]
Qinactive-Plane		Seal tipo CLR. Viene del FFD Inactive Plane (Supervisin[0.1.8]). Entra al CLR del ltimo FFD de la red de control Petri-memo. Va al bloque Processing [0.1.3.1]
Qdétection		Seal tipo CLR. Viene del FFD Détection (Supervisin[0.1.8]). Entra al CLR del ltimo FFD de la red de control Petri-vhdl-func. Va al bloque Processing [0.1.3.1]
Write-UTC		Segnal tipo SERIAL. Recibe los datos que vienen de la "UTC"[0.1.7]. Va al bloque Dater[0.1.3.2].
Write-T		Seal tipo BIT. Viene de la unidad de control. Inicializa el valor del tiempo de vuelo.
Keep-tm		Seal tipo TOKEN. Seguir tomando medidas (viene del bloque supervisor[0.1.8]). Indica que se puede continuar con el fechado del dato y su memorizacin.
Reset-address		Seal de nivel. Viene de la unidad de control.Inicializa la direccin de la memoria.
From-init		Seal tipo TOKEN. Viene de la red de control Petri-init (Modo Initialisation).
	ID-SENS [2..0]	Seal tipo DATOS. Viene del mdulo Idsensfn (bloque Logic-Nextsensor) Identificador del sensor. Viene del bloque Processing[0.1.3.1]
	clear-reg	Segnal tipo BIT. Se encarga de borrar la medida del registro en el bloque conditionner[0.1.2] para permitir la entrada de las nuevas medidas. Viene del bloque processing [0.1.3.1]
	CLR-STRT-TM	Seal tipo CLR. Enva la seal al CLR del FFD STRT-TM de la red de control Supervisor. (bloque Supervisor [0.1.8])
	ERRORSENS	Seal de ALERTA. Significa que ningn sensor se encuentra funcionando. Viene del bloque Processing [0.1.3.1]
	INACTIVE-PLANE	Seal tipo TOKEN. Va la red de control Petri-Supervisin (Supervisin).Significa que el avin se encuentra en reposo.
	DÉTECTION	Seal tipo TOKEN. Va la red de control Petri-Supervisin (Supervisin). Significa que hay un error en el sensor ya que la medida se encuentra fuera de los umbrales de funcionamiento.
	A[19..0]	Seal tipo DATOS. Corresponde a la direccin de memoria donde se deben escribir los datos. Viene del Bloque Dater [0.1.3.2].
	DQ[7..0]	Seal tipo DATOS. Corresponde a la palabra de 8 bits a ser almacenada. Viene del Bloque Dater [0.1.3.2].
	E	Seal de nivel. Ativa baja. Indica que se va a escribir en la Flash EPROM. Viene del Bloque Dater [0.1.3.2]
	G	Seal de nivel. Ativa baja. Indica que se va a escribir en la Flash EPROM. Viene del Bloque Dater [0.1.3.2]
	W	Seal de nivel. Ativa baja. Indica que se va a escribir en la Flash EPROM. Viene del Bloque Dater [0.1.3.2]
	CLR-KEEP-TM	Seal tipo CLR. Entra al CLR del ltimo FFD de la red de control Petri-Supervisor. (bloque Supervisor[0.1.8]). Viene del bloque Dater [0.1.3.2]

0.1.3.1	PROCESSING	
INPUTS	OUTPUTS	OBSERVACIONES
STRTTM		Empezar a tomar medidas (viene del bloque supervisor[0.1.8])
PRIM-DIAG[4..0]		Diagnostico de los sensores (viene del bloque tests[0.1.9])
COMMD-SENS[4..0]		Sensores a tomar medidas (viene de la Unidad de Control)
FUNC-MODE[1..0]		modo de funcionamiento (viene del supervisor[0.1.8])
write-cntr		Senal que activa la escritura en la memoria de umbrales de memorizacion (viene de la U. de control)
sensorin [7..0] MeasureIN		Medida del sensor. Viene del bloque Conditionner[0.1.2]
Qinactive-plane		Seal tipo CLR. Viene del FFD Inactive Plane (Supervisin[0.1.8]). Entra al CLR del ltimo FFD de la red de control Petri-memo. Va al bloque Processing [0.1.3.1]
Qdétection		Seal tipo CLR. Viene del FFD Détection (Supervisin[0.1.8]). Entra al CLR del ltimo FFD de la red de control Petri-vhdl-func. Va al bloque Processing [0.1.3.1]
Qdating		Seal tipo CLR. Viene de la red de control petri-flashvhdl (bloque Dating). Entra al CLR del ltimo FFD de la red de control Petri-vhdl-func.
CHANGE-THRSHLD		Seal tipo TOKEN. Viene de la red de control Petri-flashvhdl (bloque Dater[0.1.3.2]). Activa el bloque para cambiar los umbrales de memorizacin del sensor.
	CLR-STRT-TM	Seal tipo CLR. Enva la seal al CLR del FFD STRT-TM de la red de control Petri-supervisor. (bloque Supervisor[0.1.8])
	dating	Seal tipo TOKEN. Va la red de control Petri-flashvhdl (bloque Dater[0.1.3.2]). Activa el bloque encargado del fechado de la medida.
	sensorout[7..0] MeasureOUT	Seal tipo DATOS. Enva la medida tomada por el sensor actual desde el mdulo Tax8-func hacia el mdulo DAT-Unit (bloque Dater[0.1.3.2]).
	ID-SENSOR [2..0]	Seal tipo DATOS. Viene del mdulo Idsensfn (bloque Logic-Nextsensor) Identificador del sensor. Viene del bloque Processing[0.1.3.1]
	ERRORSENS	Seal de ALERTA. Significa que ningn sensor se encuentra funcionando.
	INACTIVE-PLANE	Seal tipo TOKEN. Va la red de control Petri-Supervisin (Supervisin[0.1.8]).Significa que el avin se encuentra en reposo.
	DÉTECTION	Seal tipo TOKEN. Va la red de control Petri-Supervisin (Supervisin [0.1.8]). Significa que hay un error en el sensor ya que la medida se encuentra fuera de los umbrales de funcionamiento.

0.1.3.2	DATER	
INPUT	OUTPUT	OBSERVACIONES
DATING		Seal tipo TOKEN. Va la red de control Petri-flashvhdl (bloque Dating). Activa el bloque encargado del fechado de la medida. Viene del mdulo petri-vhdl-func (bloque func-memovhd[0.1.3.1.2]).
WRITE-T		Seal tipo BIT. Viene de la unidad de control. Inicializa el valor del tiempo de vuelo.
MEASURE		Seal tipo DATOS. Viene del mdulo Tax8-Func (bloque func-memovhdl[0.1.3.1.2]). Tiene el valor de la medida tomada por el sensor para ser fechada y almacenada en la memoria.
UTC[24..4]		Seal tipo DATOS. Tiene los valores de tiempo universal. Va al mdulo Dat-Unit.
id-sensor[2..0]		Seal tipo DATOS. Viene del mdulo Idsensfn (bloque Logic-Nextsensor [0.1.3.1.1]) Identificador del sensor.
keep-tm		Seal tipo TOKEN. Seguir tomando medidas (viene del bloque supervisor [0.1.8]). Indica que se puede continuar con el fechado del dato y su memorizacin.
reset-address		Seal de nivel. Viene de la unidad de control.Inicializa la direccin de la memoria.
from-init		Seal tipo TOKEN. Viene de la red de control Petri-init (Modo Initialisation).
	CLR-KEEP-TM	Seal tipo CLR. Entra al CLR del ltimo FFD de la red de control Petri-Supervisor. (bloque Psupervisor [0.1.8]).
	CLR-DATING	Seal tipo CLR. Entra al CLR del ltimo FFD de la red de control Petri-vhdl-func. (bloque Func-memovhdl[0.1.3.1.2]).
	a[19..0]	Seal tipo DATOS. Corresponde a la direccin de memoria donde se deben escribir los datos. Va a la Flash EPROM.
	dq[7..0]	Seal tipo DATOS. Corresponde a la palabra de 8 bits a ser almacenada. Va a la Flash EPROM.
	e	Seal de nivel. Ativa baja. Indica que se va a escribir en la Flash EPROM.
	g	Seal de nivel. Ativa baja. Indica que se va a escribir en la Flash EPROM.
	change-thresholds	Seal tipo TOKEN. Va a la red de control Petri-changethrshld (bloque comp-thrshlds[0.1.3.1.3]). Indica que ya se pueden cambiar los umbrales de memorizacin.
	w	Seal de nivel. Ativa baja. Indica que se va a escribir en la Flash EPROM.

0.1.3.1.1	LOGIC-NEXTSENSOR	
INPUTS	OUTPUTS	OBSERVACIONES
STRT-TM		Empezar a tomar medidas (viene del bloque changer mode) Seal tipo TOKEN
THRIN		Seal tipo TOKEN. Viene del bloque func-memovhdl. Indica que la medida se encuentra dentro de los umbrales de memorizacin.
PRIM-DIAG[4..0]		Diagnostico de los sensores (viene del modo diagnostico) Seal tipo DATOS
CONTR-SENS[4..0]		Sensores a tomar medidas (viene de la Unidad de Control). Seal tipo DATOS
FUNC-MODE[1..0]		modo de funcionamiento (viene del modo programacion) Seal tipo DATOS
THRSHLD-MEM		Seal tipo TOKEN. Viene del bloque comp-thrshlds. Indica que ya fueron modificados los umbrales de memorizacin y se puede continuar con el sigueinte sensor.
CLR-READ-MEM		Seal tipo CLR. Viene del bloque func-memovhdl. Entra al CLR del ltimo FFD de la red de Petri-LNXT.
	ERRORSENS	Seal de ALERTA. Significa que ningn sensor se encuentra funcionando.
	IDSEN[2..0]	Seal tipo DATOS. Enva el identificador del siguiente sensor a tomar medidas. Sale hacia el bloque func-memovhdl e internamente hacia el modulo nextsen.
	CLR-THRIN	Seal tipo CLR. Enva la seal al CLR del FFD THRIN de la red de control Petri-memo. (Bloque func-memovhdl)
	CLR-THRSHLD-MEM	Seal tipo CLR. Enva la seal al CLR del FFD NEXT-SENSOR de la red de control Petri-changethrshld. (Bloque comp-thrshlds)
	CLR-STRT-TM	Seal tipo CLR. Enva la seal al CLR del FFD STRT-TM de la red de control Petri-program. (bloque Cambiar modo)
	READ-MEM	Seal tipo TOKEN. Va la red de control Petri-memo (bloque func-memovhdl). Activa el bloque encargado de la comparacin de la medida con los umbrales de memorizacin.

0.1.3.1.1.1	MULTIPL	
INPUTS	OUTPUTS	OBSERVACIONES
PRIM-DIAG[4..0]		Diagnostico de los sensores (viene del modo diagnostico)
COMMD-SENS[4..0]		Sensores a tomar medidas (viene de la Unidad de Control)
FUNC-MODE[1..0]		modo de funcionamiento (viene del modo programacion)
	sens-diag-commd[4..0]	Seal con los datos para analizar el siguiente sensor a tomar medidas. Va hacia el mdulo Firstsen y el Nextsen. Seal interna tipo DATOS.

0.1.3.1.1.2	FIRSTSEN	
INPUT	OUTPUT	OBSERVACIONES
Sens-diag-commd[4..0]		Seal interna tipo DATOS. Viene del mdulo Multpl. Recibe el estado de cada sensor.
rqst-firstsen		Seal tipo TOKEN. Activa el mdulo. Viene de la red de control Petri-Lnxt.
	ERRORSENS	Seal de ALERTA. Significa que ningn sensor se encuentra funcionando.
	first-sen[2..0]	Seal interna tipo DATOS. Va hacia el mdulo Idsensfn. Enva el identificador del primer sensor a tomar medidas.
	w1	Seal interna. Va hacia el mdulo Idesensfn. Le indica que ha llegado el identificador del primer sensor.
	ack-firstsen	Seal tipo TOKEN. Indica la terminacin de ejecucin del mdulo. Va hacia la red de control Petri-LNXT.

0.1.3.1.1.3	PETRI-LNXT	
INPUT	OUTPUT	OBSERVACIONES
STRT-TM		Empezar a tomar medidas (viene del bloque changer mode) Seal tipo TOKEN
ack-firstsen		Seal interna tipo TOKEN. Viene del mdulo Firstsen.
ack-idsen1		Seal interna tipo TOKEN. Viene del mdulo Idsensfn.
THRIN		Seal tipo TOKEN. Viene de la red de control Petri-memo (bloque fun-memovhdl).
THRSHLD-MEM		Seal tipo TOKEN. Viene de la red de control Petri-thrshld (bloque comp-thrshlds).
ack-nextsen		Seal interna tipo TOKEN. Viene del mdulo Nextsen.
CLR-READ-MEM		Seal tipo CLR. Viene del bloque func-memovhdl. Entra al CLR del ltimo FFD de la red de Petri-LNXT.
	CLR-THRSHLD-MEM	Seal tipo CLR. Enva la seal al CLR del FFD THRIN de la red de control Petri-memo. (Bloque func-memovhdl)
	CLR-THRIN	Seal tipo CLR. Enva la seal al CLR del FFD NEXT-SENSOR de la red de control Petri-changethrshld. (Bloque comp-thrshlds)
	CLR-STRT-TM	Seal tipo CLR. Enva la seal al CLR del FFD STRT-TM de la red de control Petri-program. (bloque Cambiar modo)
	rqst-firsten	Seal interna tipo TOKEN. Activa el mdulo Firstsen.
	rqst-idsen	Seal interna tipo TOKEN. Activa el mdulo Idsensfn.
	READ-MEM	Seal tipo TOKEN. Va la red de control Petri-memo (bloque func-memovhdl). Activa el bloque encargado de la comparacin de la medida con los umbrales de memorizacin.
	rqst-nextsen	Seal interna tipo TOKEN. Activa el mdulo Nextsen.

0.1.3.1.1.4	NEXTSEN	
INPUT	OUTPUT	OBSERVACIONES
sens-diag-commd[4..0]		Seal interna tipo DATOS. Viene del mdulo Multpl. Recibe el estado de cada sensor.
rqst-nextsen		Seal tipo TOKEN. Activa el mdulo. Viene de la red de control Petri-Lnxt.
idsen[2..0]		Seal interna tipo DATOS. Le indica al mdulo cual es el sensor actual para saber cual debe ser el siguiente sensor. Viene del mdulo Idsensfn.
	nextsen[2..0]	Seal interna tipo DATOS. Le indica al mdulo Idnsensfn cual es el siguiente sensor. Va hacia el mdulo Idnsensfn.
	ack-nextsen	Seal tipo TOKEN. Indica la terminacin de ejecucin del mdulo. Va hacia la red de control Petri-LNXT.

0.1.3.1.1.5	IDSENSFN	
INPUT	OUTPUT	OBSERVACIONES
w1		Seal interna. Viene del mdulo Firstsens. Le indica que ha llegado el identificador del primer sensor.
first-sen[2..0]		Seal interna tipo DATOS. Viene del mdulo Firstsenl. Recibe el identificador del primer sensor a tomar medidas.
nextsen[2..0]		Seal interna tipo DATOS. Viene del mdulo Nextsen. Recibe el identificador del siguiente sensor a tomar medidas.
rqst-idsen		Seal interna tipo TOKEN. Activa el mdulo. Viene de la red de control Petri-Lnxt.
	idsen[2..0]	Seal tipo DATOS. Enva el identificador del siguiente sensor a tomar medidas. Sale hacia el bloque func-memovhdl e internamente hacia el modulo nextsen.
	ack-idsen1	Seal interna tipo TOKEN. Indica la terminacin de ejecucin del mdulo. Va hacia la red de control Petri-LNXT.

0.1.3.1.2	FUNC-MEMO	
INPUT	OUTPUT	OBSERVACIONES
write-mem		Seal tipo TOKEN. Le indica al mdulo M5T16ACK (sub-bloque Comp-memovhdl) que se deben modificar los umbrales de un sensor en particular. Viene de la red de control Petri-Thrshld (bloque Comp-thrshlds).
write-cntr		Seal tipo TOKEN. Le indica al mdulo M5T16ACK (sub-bloque Comp-memovhdl) que se deben actualizar los umbrales de los sensores. Viene de la Unidad de control y se activa cada vez que se carguen los umbrales de memorizacin.
datain[15..0]		Seal tipo DATOS. Comprende los umbrales de memorizacin para ser modificados en el mdulo M5T16CK del sensor actual (sub-bloque comp-memovhdl). Viene del mdulo Thrshld-Modif (Bloque Comp-thrshld).
sensorin [7..0]		Seal tipo DATOS. Recibe la medida tomada por el sensor actual. Llega al mdulo Tax8Memo (sub-bloque comp-memovhdl)
id [2..0]		Seal tipo DATOS. Viene del mdulo Idsensfn (bloque Logic-Nextsensor). Identificador del sensor actual.
READ-MEM		Seal tipo TOKEN. Viene de la red de control Petri-LNXT (bloque Logic-nextsensor). Activa el bloque encargado de la comparacin de la medida con los umbrales de memorizacin. (Red de control Petri-memo)
QINACTIVE-PLANE		Seal tipo CLR. Viene del FFD Inactive Plane (Supervisin). Entra al CLR del ltimo FFD de la red de control Petri-memo.
QTHRIN		Seal tipo CLR. Viene de la red de control petri-LNXT (bloque Logic-Nextsensor). Entra al CLR del ltimo FFD de la red de control Petri-memo.
QDATING		Seal tipo CLR. Viene de la red de control petri-flashvhdl (bloque Dating). Entra al CLR del ltimo FFD de la red de control Petri-vhdl-func (sub-bloque Comp-Func-mux).
QDÉTECTION		Seal tipo CLR. Viene del FFD Détection (Supervisin). Entra al CLR del ltimo FFD de la red de control Petri-vhdl-func.
	dating	Seal tipo TOKEN. Va la red de control Petri-flashvhdl (bloque Dating). Activa el bloque encargado del fechado de la medida.
	DÉTECTION	Seal tipo TOKEN. Va la red de control Petri-Supervisin (Supervisin).Significa que hay un error en el sensor ya que la medida se encuentra fuera de los umbrales de funcionamiento.
	DATAOUT [15..0]	Seal interna tipo DATOS. Enva los umbrales de funcionamiento al comparador Tax8-func.
	sensorout[7..0]	Seal tipo DATOS. Enva la medida tomada por el sensor actual desde el mdulo Tax8-func (sub-bloque comp-func-mux) hacia el mdulo DAT-Unit (bloque Dating).
	THRIN	Seal tipo TOKEN. Va la red de control Petri-LNXT (bloque Logic-Nextsensor). Activa el mdulo Nextsen.
	ack-write	Seal tipo TOKEN. Indica la terminacin de escritura del mdulo M5T16ACK. Va hacia la red de control Petri-Thrshld.
	INACTIVE-PLANE	Seal tipo TOKEN. Va la red de control Petri-Supervisin (Supervisin).Significa que el avin se encuentra en reposo.

0.1.3.1.2.1	COMP-FUNC-MUX	
INPUTS	OUTPUTS	OBSERVACIONES
QDATING		Seal tipo CLR. Viene de la red de control petri-flashvhdl (bloque Dating). Entra al CLR del ltimo FFD de la red de control Petri-vhdl-func.
QDÉTECTION		Seal tipo CLR. Viene del FFD Détection (Supervisin). Entra al CLR del ltimo FFD de la red de control Petri-vhdl-func.
READ-FUNC		Seal interna tipo TOKEN. Viene de la red de control Petri-memo (sub-bloque comp-memovhdl). Activa el bloque encargado de la comparacin de la medida con los umbrales de funcionamiento. (Red de control Petri-funcvhdl)
sensorin[7..0]		Seal interna tipo DATOS. Recibe la medida tomada por el sensor actual del sub-bloque com-memovhdl. Llega al mdulo Tax8-func.
ID[2..0]		Seal tipo DATOS. Viene del mdulo Idsensfn (bloque Logic-Nextsensor). Identificador del sensor actual.
	dating	Seal tipo TOKEN. Va la red de control Petri-flashvhdl (bloque Dating). Activa el bloque encargado del fechado de la medida.
	sensorout[7..0]	Seal tipo DATOS. Enva la medida tomada por el sensor actual desde el mdulo Tax8-func hacia el mdulo DAT-Unit (bloque Dating).
	DATAOUT [15..0]	Seal interna tipo DATOS. Enva los umbrales de funcionamiento al comparador Tax8-func y al mdulo thrshld-comp (bloque comp-thrshld).
	DÉTECTION	Seal tipo TOKEN. Va la red de control Petri-Supervisin (Supervisin).Significa que hay un error en el sensor ya que la medida se encuentra fuera de los umbrales de funcionamiento.

0.1.3.1.2.1.1	TAX8-FUNC	
INPUTS	OUTPUTS	OBSERVACIONES
rqst-taxfunc		Seal interna tipo TOKEN. Activa el mdulo. Viene de la red de control Petri-vhdl-func.
sensorin [7..0]		Seal interna tipo DATOS. Recibe la medida tomada por el sensor actual del sub-bloque com-memovhdl.
thresholdin [15..0]		Seal interna tipo DATOS. Recibe los umbrales de funcionamiento del sensor actual. Viene del mdulo Func-mux.
	ack-thrin	Seal interna tipo TOKEN. Enva la seal de terminacin del mdulo indicando que la medida se encuentra dentro de los umbrales de funcionamiento. Va a la red de control Petri-vhdl-func.
	ack-throut	Seal interna tipo TOKEN. Enva la seal de terminacin del mdulo indicando que la medida se encuentra fuera de los umbrales de funcionamiento. Va a la red de control Petri-vhdl-func.
	sensorout [7..0]	Seal tipo DATOS. Enva la medida tomada por el sensor actual desde el mdulo Tax8-func hacia el mdulo DAT-Unit (bloque Dating).

0.1.3.1.2.1.2	PETRI-VHDL-FUNC	
INPUTS	OUTPUTS	OBSERVACIONES
READ-FUNC		Seal interna tipo TOKEN. Viene de la red de control Petri-memo (sub-bloque comp-memovhdl). Activa el bloque encargado de la comparacin de la medida con los umbrales de funcionamiento. (Red de control Petri-funcvhdl)
ACK-THROUT		Seal interna tipo TOKEN. Recibe la seal de terminacin del mdulo Tax8-func indicando que la medida se encuentra fuera de los umbrales de funcionamiento.
ACK-THRIN		Seal interna tipo TOKEN. Recibe la seal de terminacin del mdulo indicando que la medida se encuentra dentro de los umbrales de funcionamiento.
QDATING		Seal tipo CLR. Viene de la red de control petri-flashvhdl (bloque Dating). Entra al CLR del ltimo FFD de la red de control Petri-vhdl-func.
QDÉTECTION		Seal tipo CLR. Viene del FFD Détection (Supervisin). Entra al CLR del ltimo FFD de la red de control Petri-vhdl-func.
ACK-MUX		Seal interna tipo TOKEN. Indica la terminacin de seleccin de umbral de funcionamiento del sensor actual. Viene de el mdulo mux-func.
	DATING	Seal tipo TOKEN. Va la red de control Petri-flashvhdl (bloque Dating). Activa el bloque encargado del fechado de la medida.
	DÉTECTION	Seal tipo TOKEN. Va la red de control Petri-Supervisin (Supervisin). Significa que hay un error en el sensor ya que la medida se encuentra fuera de los umbrales de funcionamiento.
	READ-MUX	Seal interna tipo TOKEN. Enva la solicitud para activar el mdulo Func-mux.
	RQST-TAXFUNC	Seal interna tipo TOKEN. Enva la solicitud para activar el mdulo Tax8-func.
	CLR-READ-FUNC	Seal tipo CLR. Entra al CLR del ltimo FFD de la red de control Petri-memo.

0.1.3.1.2.1.3	MUX-FUNC	
INPUTS	OUTPUTS	OBSERVACIONES
id [2..0]		Seal tipo DATOS. Viene del mdulo Idsensfn (bloque Logic-Nextsensor). Identificador del sensor actual.
READ-MUX		Seal interna tipo TOKEN. Activa el mdulo. Viene de la red de control Petri-vhdl-func.
	DATAOUT [15..0]	Seal interna tipo DATOS. Enva los umbrales de funcionamiento al comparador Tax8-func y al mdulo thrshld-comp (bloque comp-thrshlds)
	ACK-MUX	Seal interna tipo TOKEN. Enva la terminacin de la seleccin del umbral de funcionamiento del sensor actual haca la red de control Petri-vhd-func.

0.1.3.1.2.2	COMP-MEMOVHDL	
INPUTS	OUTPUTS	OBSERVACIONES
QINACTIVE-PLANE		Seal tipo CLR. Viene del FFD Inactive Plane (Supervisin). Entra al CLR del ltimo FFD de la red de control Petri-memo.
QREAD-FUNC		Seal tipo CLR. Viene de la red de control petri-func (sub-bloque Comp-Func-Mux). Entra al CLR del ltimo FFD de la red de control Petri-memo.
QTHRIN		Seal tipo CLR. Viene de la red de control petri-LNXT (bloque Logic-Nextsensor). Entra al CLR del ltimo FFD de la red de control Petri-memo.
write-mem		Seal tipo TOKEN. Le indica al mdulo M5T16ACK que se deben modificar los umbrales de un sensor en particular. Viene de la red de control Petri-Thrshld (bloque Comp-thrshlds).
write-cntr		Seal tipo TOKEN. Le indica al mdulo M5T16ACK que se deben actualizar los umbrales de los sensores. Viene de la Unidad de control y se activa cada vez que se carguen los umbrales de memorizacin.
datain[15..0]		Seal tipo DATOS. Comprende los umbrales de memorizacin para ser modificados en el mdulo M5T16CK del sensor actual (sub-bloque comp-memovhdl). Viene del mdulo Thrshld-Modif (Bloque Comp-thrshld).
sensorin [7..0]		Seal tipo DATOS. Recibe la medida tomada por el sensor actual. Llega al mdulo Tax8Memo.
ID[2..0]		Seal tipo DATOS. Viene del mdulo Idsensfn (bloque Logic-Nextsensor). Identificador del sensor actual.
	THRIN	Seal tipo TOKEN. Va la red de control Petri-LNXT (bloque Logic-Nextsensor). Activa el mdulo Nextsen.
	ack-write	Seal tipo TOKEN. Indica la terminacin de escritura del mdulo M5T16ACK. Va hacia la red de control Petri-Thrshld.
	INACTIVE-PLANE	Seal tipo TOKEN. Va la red de control Petri-Supervisin (Supervisin).Significa que el avin se encuentra en reposo.
	READ-FUNC	Seal interna tipo TOKEN. Va a la red de control Petri-vhdl-func. Activa el bloque encargado de la comparacin de la medida con los umbrales de funcionamiento.
	sensorout[7..0]	Seal interna tipo DATOS. Enva la medida tomada por el sensor actual del sub-bloque com-memovhdl. Llega al mdulo Tax8-func.

0.1.3.1.2.2.1	TAX8MEM	
INPUTS	OUTPUTS	OBSERVACIONES
RQST-TAXMEM		Seal interna tipo TOKEN. Activa el mdulo. Viene de la red de control Petri-memo.
ID[2..0]		Seal tipo DATOS. Viene del mdulo Idsensfn (bloque Logic-Nextsensor). Identificador del sensor actual.
sensorin [7..0]		Seal tipo DATOS. Recibe la medida tomada por el sensor actual.
THRESHOLDIN		Seal interna tipo DATOS. Recibe los umbrales de memorizacin del sensor actual. Viene del mdulo M5T16ack.
	ACK-THRIN	Seal interna tipo TOKEN. Enva la seal de terminacin del mdulo indicando que la medida se encuentra dentro de los umbrales de funcionamiento. Va a la red de control Petri-memo.
	ACK-THROUT	Seal interna tipo TOKEN. Enva la seal de terminacin del mdulo indicando que la medida se encuentra fuera de los umbrales de funcionamiento. Va a la red de control Petri-memo.
	INACTIVE	Seal tipo TOKEN. Va la red de control Petri-Supervisin (Supervisin).Significa que el avin se encuentra en reposo.
	sensorout[7..0]	Seal interna tipo DATOS. Enva la medida tomada por el sensor actual del subbloque com-memovhdl. Llega al mdulo Tax8-func.

0.1.3.1.2.2.2	PETRI-MEMO	
INPUTS	OUTPUTS	OBSERVACIONES
QINACTIVE-PLANE		Seal tipo CLR. Viene del FFD Inactive Plane (Supervisin). Entra al CLR del ltimo FFD de la red de control Petri-memo.
QTHRIN		Seal tipo CLR. Viene de la red de control petri-func (subbloque Comp-Func-Mux). Entra al CLR del ltimo FFD de la red de control Petri-memo.
QREAD-FUNC		Seal tipo CLR. Viene de la red de control petri-LNXT (bloque Logic-Nextsensor). Entra al CLR del ltimo FFD de la red de control Petri-memo.
READ-MEM		Seal interna tipo TOKEN. Viene de la red de control Petri-logicnxt (bloque logic-nextsensor). Activa el bloque encargado de la comparacin de la medida con los umbrales de memorizacin. (Red de control Petri-memo)
ACK-THROUT		Seal interna tipo TOKEN. Recibe la seal de terminacin del mdulo Tax8Memo indicando que la medida se encuentra fuera de los umbrales de funcionamiento.
ACK-THRIN		Seal interna tipo TOKEN. Recibe la seal de terminacin del mdulo tax8mem indicando que la medida se encuentra dentro de los umbrales de memorizacin.
ACK-MEM		Seal interna tipo TOKEN. Rercibe la seal de terminacin del mdulo M5T16ack indicando que ya se ha leido el umbral.
INACTIVE		Seal interna tipo TOKEN. Recibe la seal de terminacin del mdulo tax8mem indicando que la medida representa que el avin est quieto o en reposo.
	INACTIVE-PLANE	Seal tipo TOKEN. Va la red de control Petri-Supervisin (Supervisin).Significa que el avin se encuentra en reposo.
	THRIN	Seal tipo TOKEN. Va la red de control Petri-LNXT (bloque Logic-Nextsensor). Activa el mdulo Nextsen.
	READ-FUNC	Seal interna tipo TOKEN. Va a la red de control Petri-vhdl-func. Activa el bloque encargado de la comparacin de la medida con los umbrales de funcionamiento.
	RQST-TAXMEM	Seal interna tipo TOKEN. Enva la solicitud para activar el mdulo Tax8mem
	CLR-READ	Seal tipo CLR. Entra al CLR del ltimo FFD de la red de control Petri-logicnxt.
	READ	Seal interna tipo TOKEN. Enva la solicitud para activar el mdulo M5T16ack en modo lectura.

0.1.3.1.2.2.3	M5T16ACK	
INPUTS	OUTPUTS	OBSERVACIONES
READ-MEM		Seal interna tipo TOKEN. Activa el mdulo. Viene de la red de control Petri-vhdl-func.
write-cntr		Seal tipo TOKEN. Le indica al mdulo M5T16ACK que se deben actualizar los umbrales de los sensores. Viene de la Unidad de control y se activa cada vez que se carguen los umbrales de memorizacin.
write-mem		Seal tipo TOKEN. Le indica al mdulo M5T16ACK que se deben modificar los umbrales de un sensor en particular. Viene de la red de control Petri-Thrshld (bloque Comp-thrshlds).
ID-MEM[2..0]		Seal tipo DATOS. Viene del mdulo Idsensfn (bloque Logic-Nextsensor). Identificador del sensor actual.
datain-MEM[15..0]		Seal tipo DATOS. Comprende los umbrales de memorizacin para ser modificados en el mdulo del sensor actual (subbloque comp-memovhdl). Viene del mdulo Thrshld-Modif (Bloque Comp-thrshld).
	DATAOUT[15..0]	Seal interna tipo DATOS. Enva los umbrales de memorizacin al comparador Tax8mem.
	ACK-MEM	Seal tipo TOKEN. Indica la terminacin de lectura del mdulo. Va hacia la red de control Petri-memo.
	ACK-WRITE	Seal tipo TOKEN. Indica la terminacin de escritura del mdulo. Va hacia la red de control Petri-Thrshld.

0.1.3.1.3	COMP-THRSHLDS	
INPUT	OUTPUT	OBSERVACIONES
sensorin [7..0]		Seal tipo DATOS. Recibe la medida tomada por el sensor actual. Llega al mdulo thrshld-mem (sub-bloque thrshld-modif). Viene del mdulo tax8-func (sub-bloque comp-func-mux).
FUNC-THRSHLD[15..0]		Seal tipo DATOS. Recibe los umbrales de funcionamiento al mdulo thrshld-comp (sub-bloque thrshld-modif). Viene del mdulo Multpl (sub-bloque comp-func-mux)
CHANGE-THRSHLD		Seal tipo TOKEN. Viene de la red de control Petri-flashvhdl (bloque Dating). Activa el bloque para cambiar los umbrales de memorizacin del sensor.
ack-write		Seal tipo TOKEN. Viene del mdulo M5T16ACK (sub-bloque comp-memovhdl). Indica la terminacin del cambio del umbral de memorizacin.
CLR-NEXT-SENS		Seal tipo CLR. Viene de la red de control petri-LNXT (bloque Logic-Nextsensor). Entra al CLR del ltimo FFD de la red de control Petri-changethrshld.
	CLR-CHANGE-THRSHLD	Seal tipo CLR. Entra al CLR del ltimo FFD de la red de control Petri-flashvhdl (bloque Dating).
	write-mem	Seal tipo TOKEN. Le indica al mdulo M5T16ACK (sub-bloque comp-memovhdl) que se deben modificar los umbrales de un sensor en particular. Va al bloque func-memovhdl.
	NEXT-SENSOR	Seal tipo TOKEN. Va al bloque Logic-Nextsensor. Indica que ya fueron modificados los umbrales de memorizacin y se puede continuar con el sigueinte sensor.
	DAT-modif[15..0]	Seal tipo DATOS. Enva al mdulo M5T16ACK (sub-Bloque Comp-memovhdl) los umbrales de memorizacin para ser modificados.

0.1.3.1.3.1	THRSHLD-MODIF	
INPUT	OUTPUT	OBSERVACIONES
sensorin [7..0]		Seal tipo DATOS. Recibe la medida tomada por el sensor actual. Llega al mdulo thrshld-mem. Viene del mdulo tax8-func (sub-bloque comp-func-mux).
modif modif		Seal interna tipo TOKEN. Activa el mdulo thrshld-mem. Viene de la red de control Petri-changethrshld.
FUNC-THRSHLD[15..0]		Seal tipo DATOS. Recibe los umbrales de funcionamiento al mdulo thrshld-comp (sub-bloque thrshld-modif). Viene del mdulo Multpl (sub-bloque comp-func-mux)
control-func		Seal interna tipo TOKEN. Activa el mdulo thrshld-comp. Viene de la red de control Petri-changethrshld.
	ack-modif	Seal tipo TOKEN. Indica la terminacin del mdulo thrshld-mem el cual hace la modificacin de los umbrales. Va hacia la red de control Petri-Thrshld.
	ack-func	Seal tipo TOKEN. Indica la terminacin del mdulo thrshld-comp el cual hace el control de los umbrales. Va hacia la red de control Petri-Thrshld.
	dat-modif [15..0]	Seal tipo DATOS. Enva al mdulo M5T16ACK (sub-Bloque Comp-memovhdl) los umbrales de memorizacin para ser modificados.

0.1.3.1.3.1.1	THRSHLD-MEM	
INPUT	OUTPUT	OBSERVACIONES
sensorin [7..0]		Seal tipo DATOS. Recibe la medida tomada por el sensor actual. Viene del mdulo tax8-func (sub-bloque comp-func-mux).
modif		Seal interna tipo TOKEN. Activa el mdulo thrshld-mem. Viene de la red de control Petri-changethrshld.
	ACK	Seal tipo TOKEN. Indica la terminacin del mdulo thrshld-mem el cual hace la modificacin de los umbrales. Va hacia la red de control Petri-Thrshld.
	DATAIN[15..0]	Seal tipo DATOS. Enva al mdulo thrshld-comp los umbrales de memorizacin modificados para compararlos con los umbrales de funcionamiento.

0.1.3.1.3.1.2	THRSHLD-COMP	
INPUT	OUTPUT	OBSERVACIONES
data[15..0]		Seal interna tipo DATOS. Recibe del mdulo thrshld-mem los umbrales de memorizacin modificados para compararlos con los umbrales de funcionamiento.
FUNC-THRSHLD[15..0]		Seal tipo DATOS. Recibe los umbrales de funcionamiento al mdulo thrshld-comp (sub-bloque thrshld-modif). Viene del mdulo Multpl (sub-bloque comp-func-mux)
control-func		Seal interna tipo TOKEN. Activa el mdulo thrshld-comp. Viene de la red de control Petri-changethrshld.
	ack-thrshldcomp	Seal tipo TOKEN. Indica la terminacin del mdulo thrshld-mem el cual hace la modificacin de los umbrales. Va hacia la red de control Petri-Thrshld.
	dat-modif[15..0]	Seal tipo DATOS. Enva al mdulo M5T16ACK (sub-Bloque Comp-memovhdl) los umbrales de memorizacin para ser modificados.

0.1.3.1.3.2	PETRI-CHANGETHRSHLD	
INPUT	OUTPUT	OBSERVACIONES
CHANGE-THRSHLD		Seal tipo TOKEN. Viene de la red de control Petri-flashvhdl (bloque Dating). Activa el bloque para cambiar los umbrales de memorizacin del sensor.
CLR-NEXT-SENS		Seal tipo CLR. Viene de la red de control petri-LNXT (bloque Logic-Nextsensor). Entra al CLR del ltimo FFD de la red de control Petri-changethrshld.
ACK-MODIF		Seal tipo TOKEN. Indica la terminacin del mdulo thrshld-mem el cual hace la modificacin de los umbrales.
ACK-CONTROL		Seal tipo TOKEN. Indica la terminacin del mdulo thrshld-mem el cual hace la modificacin de los umbrales.
ack-write		Seal tipo TOKEN. Viene del mdulo M5T16ACK (sub-bloque comp-memovhdl). Indica la terminacin del cambio del umbral de memorizacin.
	CLR-CHANGE-THRSHLD	Seal tipo CLR. Entra al CLR del ltimo FFD de la red de control Petri-flashvhdl (bloque Dating).
	NEXT-SENSOR	Seal tipo TOKEN. Va al bloque Logic-Nextsensor. Indica que ya fueron modificados los umbrales de memorizacin y se puede continuar con el sigueinte sensor.
	MODIF	Seal interna tipo TOKEN. Activa el mdulo thrshld-mem.
	CONTROL-FUNC	Seal interna tipo TOKEN. Activa el mdulo thrshld-comp.
	write-mem	Seal tipo TOKEN. Le indica al mdulo M5T16ACK (sub-bloque comp-memovhdl) que se deben modificar los umbrales de un sensor en particular. Va al bloque func-memovhdl.

0.1.3.2.1	DATE-UNIT	
INPUTS	OUTPUTS	OBSERVACIONES
rqst-date		Seal interna tipo TOKEN. Activa el mdulo. Viene de la red de control Petri-flahsvhdl.
write-t		Seal tipo BIT. Viene de la unidad de control. Inicializa el valor del tiempo de vuelo.
utc[24..4]		Seal tipo DATOS. Tiene los valores de tiempo universal.
id-sensor[2..0]		Seal tipo DATOS. Viene del mdulo Idsensfn (bloque Logic-Nextsensor). Identificador del sensor actual.
measure		Seal tipo DATOS. Viene del mdulo Tax8-Func (bloque func-memovhdl). Tiene el valor de la medida tomada por el sensor para ser fechada y almacenada en la memoria.
	byte1[7..0]	Seal interna tipo DATOS. Va al mdulo W1-Eprom para ser puesto en la memoria EPROM. Contiene el tiempo de vuelo.
	byte2[7..0]	Seal interna tipo DATOS. Va al mdulo W1-Eprom para ser puesto en la memoria EPROM. Contiene el identificador del sensor [2..0] y los valores menos significativos de la UTC [8..4].
	byte3[7..0]	Seal interna tipo DATOS. Va al mdulo Write-Eprom para ser puesto en la memoria EPROM. Contiene los valores ms significativos de la UTC [16..9].
	ack-date	Seal tipo TOKEN. Indica la terminacin del mdulo el cual acomoda los datos a almacenar en 3 bytes. Va hacia la red de control Petriflash-vhdl.
	begin-write	Seal interna. Va al mdulo Write-eprom y es la que le indica que se debe escribir el Byte 1 en la Eprom.

0.1.3.2.2	WRITE-EPROM	
INPUT	OUTPUTS	OBSERVACIONES
rqst-w1		Seal interna tipo TOKEN. Activa el mdulo. Viene de la red de control Petri-flahsvhdl.
begin-write		Seal interna. Viene del mdulo Dat-unit y es la que le indica que se debe escribir el Byte 1 en la Eprom.
byte1[7..0]		Seal interna tipo DATOS. Viene del mdulo Dat-unit. Para se puesto en la memoria EPROM. Contiene el tiempo de vuelo.
byte2[7..0]		Seal interna tipo DATOS. Viene del mdulo Dat-unit para ser puesto en la memoria EPROM. Contiene el identificador del sensor [2..0] y los valores menos significativos de la UTC [8..4].
byte3[7..0]		Seal interna tipo DATOS. Viene del mdulo Dat-unit para ser puesto en la memoria EPROM. Contiene los valores ms significativos de la UTC [16..9].
	ack-again	Seal tipo TOKEN. Indica la terminacin del mdulo pero que debe enviar el siguiente Byte para ser puesto en la memoria. Va hacia la red de control Petriflash-vhdl.
	ack-end	Seal tipo TOKEN. Indica la terminacin del mdulo, todos los bytes ya han sido puestos en la memoria y se puede continuar con el ajuste de los umbrales de memorizacin. Va hacia la red de control Petriflash-vhdl.
	dq[7..0]	Seal tipo DATOS. Corresponde a la palabra de 8 bits a ser almacenada. Va a la Flash EPROM.

0.1.3.2.3	PETRIFLASH-VHDL	
INPUT	OUTPUTS	OBSERVACIONES
keep-tm		Seal tipo TOKEN. Seguir tomando medidas (viene del bloque changer mode). Indica que se puede continuar con el fechado del dato y su memorizacin.
dating		Seal tipo TOKEN. Va la red de control Petri-flashvhdl (bloque Dating). Activa el bloque encargado del fechado de la medida. Viene del mdulo petri-vhdl-func (bloque func-memovhdl).
from-init		Seal tipo TOKEN. Viene de la red de control Petri-init (Modo Initialisation).
ack-date		Seal tipo TOKEN.Viene del mdulo dat-unit.
ack-again		Seal tipo TOKEN. Indica la terminacin del mdulo pero que debe enviar el siguiente Byte para ser puesto en la memoria. Viene del mdulo Write-Eprom.
ack-end		Seal tipo TOKEN. Indica la terminacin del mdulo, todos los bytes ya han sido puestos en la memoria y se puede continuar con el ajuste de los umbrales de memorizacin. Viene del mdulo Write-Eprom.
ack-contaddr		Seal tipo TOKEN. Indica la terminaci.on del mdulo Contaddr, encargado de actualizar la siguiente posicin de memoria.
ack-regaddr		Seal tipo TOKEN. Indica la terminacin del mdulo Regaddr, encargado de enviar la direccin de memoria correspondiente.
	CLR-KEEP-TM	Seal tipo CLR. Entra al CLR del ltimo FFD de la red de control Petri-program. (bloque Program).
	CLR-DATING	Seal tipo CLR. Entra al CLR del ltimo FFD de la red de control Petri-vhdl-func. (bloque Func-memovhdl).
	rqst-date	Seal interna tipo TOKEN. Activa el mdulo Dat-unit.
	rqst-w1	Seal interna tipo TOKEN. Activa el mdulo Write-Eprom.
	rqst-contaddr	Seal interna tipo TOKEN. Activa el mdulo Contaddr.
	rqst-regaddr	Seal interna tipo TOKEN. Activa el mdulo Regaddr.
	change-thresholds	Seal tipo TOKEN. Va a la red de control Petri-changethrshld (bloque comp-thrshlds). Indica que ya se pueden cambiar los umbrales de memorizacin.

0.1.3.2.4	CONTADDRESS	
INPUTS	OUTPUTS	OBSERVACIONES
rqst-contaddr		Seal interna tipo TOKEN. Activa el mdulo. Viene de la red de control Petri-flahsvhdl.
reset-address		Seal de nivel. Viene de la unidad de control.Inicializa la direccin de la memoria.
	ack-contddr	Seal tipo TOKEN. Indica la terminacion del mdulo. Va a la red de control Petriflash-vhdl encargado de actualizar la siguiente posicin de memoria.
	address[19..0]	Seal interna tipo DATOS. Enva la direccin de memoria correspondiente al mdulo Regaddr.

0.1.3.2.5	REGADDRESS	
INPUT	OUTPUT	OBSERVACIONES
rqst-regaddr		Seal interna tipo TOKEN. Activa el mdulo. Viene de la red de control Petri-flahsvhdl.
address[19..0]		Seal interna tipo DATOS. Viene del mdulo Contaddr. Contiene la direccin de memoria correspondiente.
	ack-regaddr	Seal tipo TOKEN. Indica la terminacion del mdulo. Va a la red de control Petriflash-vhdl.
	a[19..0]	Seal tipo DATOS. Corresponde a la direccin de memoria donde se deben escribir los datos. Va a la Flash EPROM.

Table des figures

Bibliographie

[1] D. ESTEVE, J. BOUCHER, D. ANDREU, J.N. CONTENSOU, N. HARCHANI, M. COURVOISIER, A. GARCIA, F. JIMENEZ, M. POLLINA. M3: PLATEFORME DE TÉLÉCONCEPTION DE SYSTÈMES MICROÉLECTRONIQUES. IEEE Symposium on Education and Employment, ISCEE'98, Amiens, France, Octobre 1998.

[2] F. JIMENEZ, M. COURVOISIER, A. GARCIA, D. ESTEVE. MODELS FOR RAPID PROTOTYPING OF REAL TIME EMBEDDED SYSTEMS BASED ON ASYNCHRONOUS SYSTEMS. KIT 106 Workshop on Integrated Systems on Silicon: Design Methodologies and Case Studies, Brasil, December 1998. Sponsor: European Comitee - Keep In Touch Program KIT 106. Parners: IMEC (Belgium), TIMA (France), LME-EPUSP (Brasil).

[3] A. GARCIA, G. MUNOZ, M. COURVOISIER, F. JIMENEZ. SDL TOOLS FOR ARCHITECTURAL DESIGN OF ASYNCHRONOUS DIGITAL SYSTEMS. V International Workshop Iberchip, Peru, Mars 1999

[4] F. JIMENEZ, M. COURVOISIER, A. GARCIA, G. MUNOZ. N. HARCHANI, D. ESTEVE, M. AL-MOHAMED. TOOLS AND MODELS FOR SYSTEM DESIGN AND SYNTHESIS OF MEMS BASED ON ASYNCHRONOUS CIRCUITS. Proceedings of the IEEE International Conference on Industrial Technology, Goa, India, 19-22 January 2000.

[5] N. HARCHANI, F. JIMENEZ, M. AL-MOHAMED, D. ESTEVE, M. COURVOISIER. TIME STRESS MEASUREMENT DEVICE - SYSTEM DESIGN AND SYNTHESIS. International Society for Optical Engineering SPIE AeroSense 2000, Orlando, Florida, 24-28 April 2000.

[6] N. HARCHANI, F. JIMENEZ, M. AL-MOHAMED, D. ESTEVE, M. COURVOISIER. DEVELOPMENT OF A SPECIFIC CAD ENVIRONMENT FOR A TOP DOWN DESIGN OF MEMS. , 16th International Conference on CAD/CAM, Robotics and Factories of the Future, CARS AND FOF 2000, Port Spain, Trinidad W.I., 26-28 June, 2000.

[7] N. HARCHANI, F. JIMENEZ, D. ESTEVE, J. BOUCHER, D. ANDREU, J.N. CONTENSOU, M. COURVOISIER, M. POLLINA, A. GARCIA, A NEW EDUCATIONAL APPROACH IN THE MICROSYSTEMS WORD., MIXDES 99, Krakow, Poland, June 1999.

[8] F. JIMENEZ, N.HARCHANI, AL-MOHAMED, M.COURVOISIER, D. ESTEVE. A STRUCTURED METHODOLOGY FOR THE DESIGN OF AUTONOMOUS MULTISENSORS CIRCUITS, IFAC 4th Symposium SICICA 2000, Buenos Aires, Argentina, 13-15 September 2000

[9] R. VALETTE, M. COURVOISIER, SYSTÈMES DE COMMANDE TEMPS RÉEL, Editions SCM, 1980, Paris

[10] N. HARCHANI, "ETUDE D'UNE MÉTHODOLOGIE DE CONCEPTION DESCENDANTE DES MICRO SYSTÈMES: CONCEPTION D'UN MICRO SYSTÈME POUR LA SURVEILLANCE DES CONTRAINTES MÉCANIQUES EN AÉRONAUTIQUE", These, ENSEEIHT, Toulouse, 2000

[11] INTERNATIONAL TECHNOLOGY ROADMAP FOR SEMICONDUCTORS, http://notes.sematech.org/ntrs/Rdmpmem.nsf

[12] D. BRAHA, O. MAIMOM, "THE DESIGN PROCESS: PROPERTIES, PARADIGMS, AND STRUCTURE", IEEE Transactions, on Man, Systems and Cybernetics - Part A, Vol. 27, NO. 2, 1997

[13] S. BEN AHMED, "APPROCHE MULTI MODELES POUR LA SPECIFICATION, LA VALIDATION ET LA CONCEPTION DE LA COMMANDE DES SYSTEMES FELXIBLES DE PRODUCTION MANUFACTURIERE", THESE, Université des Sciences, des Techniques et de Médicine de Tunis, 1997

[14] P.WARD, S.MELLOR, "STRUCTURED DEVELOPMENT FOR REAL TIME SYSTEMS", 1985

[15] D.HATLEY, A. PIRBHAI "STRATÉGIES FOR REAL TIME SYSTEMS", 1987

[16] M. COMBACAU, "COMMANDE ET SURVEILLANCE DES SYSTEMES A EVENEMENTS DISCRETS COMPLEXES: APPLICATION AUX ATELIERS FLEXIBLES", Thèse Doctorat, Université Paul Sabatier, Toulouse, 1991

[17] E. ZAMAI, "ARCHITECTURE DE SURVEILLANCE 6 COMMANDE POUR LES SYSTEMES A EVENEMENTS DISCRETS COMPLEXES", Thèse Doctorat, Université Paul Sabatier, Toulouse, 1997

[18] A. CHAILLET, "APPROCHE MULTIMODELES POUR LA COMMANDE ET LA SURVEILLANCE EN TEMPS REEL DES SYSTEMES A EVENEMENTS DISCRETS", Thèse Doctorat, Université Paul Sabatier, Toulouse, 1995

[19] R.M WALTERS ,D.A BRADLEY, A.P DOREY, "A CONCEPTUAL STUDY FOR A COMPUTER-BASED TOOL SUPPORT ELECTRONIC DESIGN IN

A MECHATRONIC ENVIRONMENT",Microprocessors and Microsystems, Vol 24, Num 2, April 2000

[20] A.JERRAYA, K.O'BRIEN "SOLAR: AN INTERMEDIATE FORMAT FOR SYSTEM LEVEL MODELING AND SYNTHESIS", Codesign: Computer Aided Software/Hardware Engineering, IEEE Press, 1995, Chapter 7

[21] N.MARRANGHELLO, J.MIRKOWSKI, K.BILINSKI, "SYNTHESIS OF SYNCHRONOUS DIGITAL SYSTEMS SPECIFIED BY PETRI NETS" Workshop Hardware Design and petri nets, HWPN'98, June 23, 1998, Lisbon, Portugal

[22] D.GARETH EVENS, D.MORRIS, "APPLYING MODELLING TO EMBEDDED COMPUTER SYSTEM DESIGN", Codesign: Computer Aided Software/Hardware Engineering, IEEE Press, 1995, Chapter 4

[23] P.ROKYTA, W.FENGLER, T.HUMMEL, "ELECTRONIC SYSTEM DESIGN AUTOMATION USING HIGH LEVEL PETRI NETS", 19 International Conference on Application and Theory of Petri Nets", 1998, Lisbon, Portugal

[24] D.WIKARSKY (Ed), "PETRI NET TOOLS: A COMPARATIVE STUDY", ISST-Bericht 39/96 of the Fraunhofer-Gesellschaft e.G., Berlin, 1996

[25] H.STORRLE, "AN EVALUATION OF HIGH END TOOLS FOR PETRI NETS", Institut fur Informatik, LMU, Munchen, stoerrle@informatik.uni-muenchen.de, v 1.1, 1998

[26] S.TAN, S.FURBER, W.YEN, "THE DESIGN OF AN ASYNCHRONOUS VHDL SYNTHESIZER", Proceedings of Design, Automation and Test in Europe, Paris, February 1998.

[27] C.GRIM, K.WALDSCHMIDT, "REPARTITIONING AND TECHNOLOGY MAPPING OF ELECTRONIC HYBRID SYSTEMS", Proceedings of Design, Automation and Test in Europe, Paris, February 1998.

[28] E.MOSER, N.MITTWOLLEN, "VHDL-AMS: THE MISSING LINK IN SYSTEM DESIGN - EXPERIMENTS WITH UNIFIED MODELLING IN AUTOMOTIVE ENGINEERING", Proceedings of Design, Automation and Test in Europe, Paris, February 1998.

[29] S.VERCAUTEREN, D.VERKEST, G.JONG, B.LIN, "DERIVATION OF FORMAL REPRESENTATION FROM PROCESS-BASED SPECIFICATION AND IMPLEMENTATION MODELS", The Proceedings of the International Symposium on System Synthesis, IEEE, Belgium, 1997.

[30] E.PASTOR, J.CORTADELLA, "EFFICIENT ENCODING SCHEMES FOR SYMBOLIC ANALYSIS OF PETRI NETS", Proceedings of Design, Automation and Test in Europe, Paris, February 1998.

[31] J.CORTADELLA, L.LAVAGNO, P.VANBEKBERGEN, A.YAKOVLEV, "DESIGNING ASYNCHRONOUS CIRCUITS FROM BEHAVIOURAL SPECIFICATIONS WITH INTERNAL CONFLICTS", UPC/DAC Technical report No. RR94/08

[32] J.CORTADELLA, M.KISHINEVSKY, A.KONDRATYEV, L.LAVAGNO, "COMPLETE STATE ENCODING BASED ON THEORY OF REGIONS", Proceedings of ASYNC'96, pp. 36-47, Aizu, Japan, March, 1996

[33] M.KASSAB, E.CERNY, S.AOURID, T.KRODEL, "PROPAGATION OF LAST TRANSITION TIME CONSTRAINS IN GATE LEVEL TIMING ANALYSIS", Proceedings of Design, Automation and Test in Europe, Paris, February 1998.

[34] M.MRVA, K.BUCHENRIEDER, R.KRESS, "A SCALABLE ARCHITECTURE FOR MULTI-THREADED JAVA APLICATIONS, Proceedings of Design, Automation and Test in Europe, Paris, February 1998.

[35] M.MUTZ, "REGISTER TRANSFER LEVEL VHDL MODELS WITHOUT CLOCKS", Proceedings of Design, Automation and Test in Europe, Paris, February 1998.

[36] M.LUBASZEWSKI, E.COTA, B.COURTOIS, "MICROSYSTEMS TESTING: AN APPROACH AND OPEN PROBLEMS", Proceedings of Design, Automation and Test in Europe, Paris, February 1998.

[37] A.GARCIA, "ARQUITECTURA DE SISTEMAS ELECTRONICOS", Capitulo Primero, Uniandes, Colombia, 1998

[38] R.WOLLOWSKY, J.BEISTER, "PRECISE PETRI NET MODELLING OF CRITICAL RACES IN ASYNCHRONOUS ARBITERS AND SYNCHRONIZERS", 19 International Conference on Application and Theory of Petri Nets", 1998, Lisbon, Portugal

[39] F.XIA, I.CLARK, "COMPLEMENTING ROLE MODELS WITH PETRI NETS IN STUDYING ASYNCHRONOUS DATA COMMUNICATIONS", 19 International Conference on Application and Theory of Petri Nets", 1998, Lisbon, Portugal

[40] A.GIRAULT, B.LEE, E.LEE, "HIERARCHICAL FINITE STATE MACHINES WITH MULTIPLE CONCURRENCY MODELS", University of California at San Francisco, Octobre 19 1998, revised from Technical Memorandum UCB/ERL M97/57.

[41] M.KISHINEVSKY, J.CORTADELLA, A.KONDRATYEV, "ASYNCHRONOUS INTERFACE SPECIFICATION, ANALYSIS AND SYNTHESIS", Proceedings of the 35th Design Automation Conference, San Francisco, CA, June 1998.

[42] J.DEDOU, O.SENTIEYS, J.L.PHILIPPE, "SYNTHESE D'ARCHITECTURES ASYNCHRONES EN TRAITEMENT NUMERIQUE DU SIGNAL", Colloque CAO de circuits intégrés et systèmes, DSPT4/MENESRIP, Grenoble, Janvier 1997.

[43] D.GAJSKI, F.VAHID, S.NARAYAN, J.GONG, "SYSTEM LEVEL EXPLORATION WITH SPECSYN", Proceedings of the 35th Design Automation Conference, San Francisco, CA, June 1998.

[44] LUCKHAM, DAVID " THREE CONCEPTS OF SYSTEM ARCHITECTURE " [en ligne]. The Stanford Rapide Project, Rapide Publications, Program Analysis and Verification Group, Computer Science Department, Electrical Engineering Department, Stanford University, November 1999, http://pavg.stanford.edu/rapide/rapide-pubs.html [Consulté le 14 novembre 99]

[45] MISTREE, F. LEARNING HOW TO DESIGN: A MINDS-ON, HANDS-ON, DECISION-BASED APPROACH [on line] Georgia Institute of Technology, School of Mechanics Engineering. 1995. http://www.srl.gatech.edu/education/ME3110 [Consulted: November 24 1999]

[46] MILLER, J.G. LIVING SYSTEMS, McGraw-Hill Book Company, New York, 1996

[47] ANSI/EIA STANDARD 632-1998 PROCESS FOR ENGINEERING A SYSTEM

[48] A. DEWEY, H. DUSSAULT and J. HANNA, E. CHRISTEN, G. FEDDER, B. ROMANOWICZ and M. MAHER, ENERGY-BASED CHARACTERIZATION OF MICROELECTROMECHANICAL SYSTEMS (MEMS) AND COMPONENT MODELING USING VHDL-AMS, Modeling and Simulation of Microsystems, Motorola, IEEE Electron Devices Society, San Juan Puerto Rico, USA, Computational Publications, Vol pp. 139-142, April 19-21, 1999

[49] AL, RACHID GUERRAQUI ET, STRATEGIC DIRECTIONS IN OBJECT-ORIENTED PROGRAMMING, ACM Computing Surveys, Vol 28, Issue 4, pp. 691-700, December 1996

[50] AL DAVIS, STEVEN M. NOWICK, ASYNCHRONOUS CIRCUIT DESIGN: MOTIVATIONS, BACKGROUND, AND METHODS, Asynchronous Digital Circuit Design, Ed G. Birttwistle, A. Davis, Springler, London, Vol Ed pp. 1-49, 1995

[51] ALEX KONDRATYEV, JORDI CORTADELLA, MICHAEL KISHINEVSKY, LUCIANO LAVAGNO, ALEXANDRE YAKOVLEV, LOGIC DECOMPOSITION OF SPEED-INDEPENDENT CIRCUITS, PROCEEDINGS OF THE IEEE, Vol 87, Issue 2, pp. 347-362, FEBRUARY 1999

[52] ALEX SEMENOV, ALBERT M. KOELMANS, LEE LLOYD and ALEXANDRE YAKOVLEV, DESIGNING AN ASYNCHRONOUS PROCESSOR USING PETRI NETS, IEEE Micro, Vol Issue pp. 54-64, March/April, 1997

[53] BERKS, R., PERFORMANCE ANALYSIS OF ASNCHRONOUS NETWORKS, Thesis of the University of Waterloo, July, 1998

[54] BORRIELLO, KEN HINES AND GAETANO, A GEOGRAPHICALLY DISTRIBUTED FRAMEWORK FOR EMBEDDED SYSTEM DESIGN AND VALIDATION, San Francisco, California, ACM, Vol pp. 6, 1998

[55] CHO W. MOON, PAUL R. STEPHAN, ROBERT K. BRAYTON, SYNTHESIS OF HAZARD-FREE ASYNCHRONOUS CIRCUITS FROM GRAPHICAL SPECIFICATIONS, IEEE International Conference on Compute-Aided Design, Santa Clara - California, IEEE Computer Society Press, Vol pp. 322-325, November 11-14, 1991

[56] CORTADELLA, MARCO A. PENA and JORDI, COMBINING PROCESS ALGEBRAS AND PETRI NETS FOR THE SPECIFICATION AND SYNTHESIS OF ASYNCHRONOUS CIRCUITS, Second International Symposium on Advanced Research in Asynchronous Circuits and Systems, Aizu-Wakamatsu, Fukushima, Japan, IEEE Computer Society, Vol pp. 222-232, March 18-21, 1996

[57] CORTADELLA, ENRIC PASTOR and JORDI, POLYNOMIAL ALGORITHMS FOR THE SYNTHESIS OF HAZARD-FREE CIRCUITS FROM SIGNAL TRANSITION GRAPHS, IEEE/ACM International Conference on Computer-Aided Design, Santa Clara, California, IEEE Computer Society, Vol pp. 250-254, November 7-11, 1993

[58] D.W. LLOYD, J.D. GARSIDE, D.A. GILBERT, MEMORY FAULTS IN ASYNCHRONOUS MICROPROCESSORS, Advanced Research in Asynchronous Circuits and Systems: Microprocessor Design, Society, IEEE Computer, Barcelona, Spain, Los Alamitos, California, Vol pp. 71-80, April 19-21, 1999

[59] DAVIS, AL, SYNTHESIZING ASYNCHRONOUS CIRCUITS: PRACTICE AND EXPERIENCE, Asynchronous Digital Circuit Design, Ed G. Birttwistle, A. Davis, Springler, London, Vol Ed pp. 104-150, 1995

[60] E. GRASS, R. C. S. MORLING and I. KALE, ACTIVITY-MONITORING COMPLETION-DÉTECTION (AMCD): A NEW SINGLE RAIL APPROACH TO ACHIEVE SELF-TIMING, Second International Symposium on Advanced Research in Asynchronous Circuits and Systems, Aizu-Wakamatsu, Fukushima, Japan, IEEE Computer Society, Vol pp. 143-151, March 18-21, 1996

[61] HENRIK HULGAARD, STEVEN M. BURNS and GAETANO BORRIELLO, TESTING ASYNCHRONOUS CIRCUITS: A SURVEY, Thesis of the University of Washington, March 6, 1994

[62] HUNT, A. COOK and K.J.R., ARINC 653 - ACHIEVING SOFTWARE RE-USE, Microprocessors and Microsystems, Vol 20, Issue pp. 479-483, 1997

[63] J. BEISTER, G. ECKSTEIN and R. WOLLOWSKI, FROM STG TO EXTENDED-BURST-MODE MACHINES, Advanced Research in Asynchronous Circuits and Systems: Synthesis, Society, IEEE Computer, Barcelona, Spain, Los Alamitos, California, Vol pp. 145-158, April 19-21, 1999

[64] JOEP KESSELS, PAUL MARSTON, DESIGNING ASYNCHRONOUS STANDBY CIRCUITS FOR A LOW-POWER PAGER, PROCEEDINGS OF THE IEEE, Vol 87, Issue 2, pp. 257-267, FEBRUARY 1999

[65] KHODABANDEHLOO, KOOROSH, ANALYSES OF ROBOT SYSTEMS USING FAULT AND EVENT TREES: CASE STUDIES, Reliability Engineering and System Safety, Vol 53, Issue pp. 247-264, 1996

[66] KLAUS ACHOSSMAIER, MARTIN HORAUER, DIETMAR LOY,, SPECIFICATION AND IMPLEMENTATION OF THE UNIVERSAL TIME COORDINATED SYNCHRONIZATION UNIT (UTCSU), Real-Time Systems. (c) 1997 Kluwer Academic Publishers, Boston. Vol 12, Issue pp. 295-327, 1997

[67] MARTIN, GRANT, DESIGN METHODOLOGIES FOR SYSTEM LEVEL IP, Design, Automation and Test in Europe, Pars - Francia, IEEE Computer Society, Vol pp. 286-289, February 23-24, 1998

[68] MICHAEL PECHT, PETER SANDBORN, and PATRICK McCLUSKEY, VIRTUAL COMPONENT QUALIFICATION, Modeling and Simulation of Microsystems, Motorola, IEEE Electron Devices Society, San Juan Puerto Rico, USA, Computational Publications, Vol pp. 8-13, April 19-21, 1999

[69] SACHS, MARK BIRNBAUM and HOWARD, HOW VSIA ANSWERS THE SOC DILEMA, Computer Innovative technology for computer professionals, Vol 32, Issue 6, pp. 42-50, June, 1999

[70] SALEFSKI, GRANT MARTIN and BILL, METHODOLOGY AND TECHNOLOGY FOR DESIGN OF COMMUNICATIONS AND MULTIMEDIA PRODUCTS VIA SYSTEM-LEVEL IP INTEGRATION, Design, Automation and Test in Europe, France, IEEE Computer Society, Vol pp. 11-18, 23-26, 1998

[71] SILCOTT, GARY, SOC S DRIVE NEW PRODUCT DEVELOPMENT, Computer Innovative technology for computer professionals, Vol 32, Issue 6, pp. 61-66, June, 1999

[72] STAHR CARLOS, THOMAS ANDERSON, ASIC DESIGN FLOW WITH SYNTHESIZABLE CORES, Design, Automation and Test in Europe, France, IEEE Computer Society, Vol pp. 137-139, 23-26, 1998

[73] STEVEN M. NOWICK, KENNETH Y. YUN, PETER A. BEEREL and AYOOB E. DOOPLY, SPECULATIVE COMPLETION FOR THE DESIGN OF HIGH-PERFORMANCE ASYNCHRONOUS DYNAMIC ADDERS, Proceedings of the IEEE Third International Symposium on Advanced Research in Asynchronous Circuits and Systems, Eindhowen, The Netherlands, 1997

TITRE DE LA THÈSE EN FRANÇAIS :

Specification et conception de microsystèmes bases sur des circuits asynchrones - Étude d'un dispositif multicapteur integré d'enregistrement de contraintes environnementales.

RESUMÉ EN FRANCAIS:

L'introduction des systèmes électroniques embarqués et miniaturisés dans des nouveaux dispositifs apporte des avantages qui sont ressentis essentiellement comme une réduction des coûts réels et une augmentation de la performance. Les impacts les plus importants concernent la réduction de la complexité mécanique (réduction du câblage par exemple), une meilleure opération (par l'augmentation de la flexibilité des fonctionnalités) et une maintenance plus efficace (par une meilleure connaissance de l'état de vieillissement du système). L'électronique étant maîtrisée, la compétitivité tient dans une conception, qui soit rapide, sans faute, robuste et réalisable. Les efforts doivent s'orienter vers la recherche de méthodologies et techniques qui puissent supporter l'intégration, principal défi proposé par la conception de ces nouveaux systèmes répartis, communicants et pluridisciplinaires. Cette thèse presente les modèles et les outils developpés pour aider à la conception de Micro Systèmes, avec des points forts tels que: la conception amont, les systèmes asynchrones, la fiabilité et l'intégration au système de Conception Structurée Descendante. Ces modèles et outils sont basés sur SDL, SA-RT et les Réseaux de Petri.

MOTS-CLES :

Conception de Microsystèmes, Systèmes et circuits asynchrones, Fiabilité des Systèmes Autonomes, Conception Structurée Descendante, SDL, SA-RT, Réseaux de Petri.

TITRE DE LA THÈSE EN ANGLAIS :

System Design and Synthesis of Micro Systems based on Asynchronous Circuits - System Design and Synthesis of a Time Stress Measurement Device.

RÉSUMÉ DE LA THÈSE EN ANGLAIS :

The improvement of micro-technologies in the microelectronics world has permitted the creation of ultra miniaturised devices, which are called MEMS – *Micro Electro Mechanical Systems.* A major problem concerns the design of these heterogeneous circuits because it associates disciplines such as electronics, mechanics, chemistry, etc. This thesis presents models and tools developed for System Design and Synthesis of MEMS based on SDL (Specification Description Language), Structured Analysis for Real Time (SA-RT) and Petri Nets (PN).

MOTS CLES EN ANGLAIS:

Microsystems Design, Asynchronous Systems and circuits, Reliability of Autonomous Systems, Top-down Structured Design, SDL, SA-RT, Petri nets.