

# Méthodologie de conception à base d'IPs

Ahcene Youcef Benabdallah , Rachid Boudour  
Dept d'informatique Univ. Badji-Mokhtar  
Annaba, Algérie  
Benayouc@Hotmail.fr

## Abstract:

*Poussés par les contraintes de " time to market " et de complexité galopante des systèmes embarqués principalement des systèmes sur puce ou SoC, les concepteurs de compétences différentes, sont amenés à réutiliser de plus en plus de composants virtuels ou Intellectual Properties (IPs). Dans ce sillage, nous présentons une définition d'un cadre pour la spécification et la réutilisation d'IPs. Pour ce faire, nous avons adopté une démarche au niveau comportemental permettant la conception de l'IP et la création d'une bibliothèque, accessible en vue de la sélection, le paramétrage et l'intégration des composants dans des applications. Des exemples ont été expérimentés pour valider nos propos.*

**Keywords-:** Réutilisation, IPs , SoCs , Bibliothèque IPs

## 1. Introduction

L'évolution rapide de notre société conduit inéluctablement à l'élaboration d'équipements de plus en plus complexes, capables de traiter des flots d'informations de nature et d'origine très diverses et dont l'accès doit néanmoins rester simple et rapide. La conception et l'intégration de ce type d'équipements (systèmes) demande aujourd'hui encore des développements longs, fastidieux, coûteux et par conséquent en inadéquation avec les évolutions rapides et la flexibilité que doivent suivre les matériels électroniques. Ajouter à cela, le fait qu'il est aujourd'hui très difficile de réutiliser la connaissance existante, l'industrie se trouve confrontée à un problème sérieux : le manque de concepteurs de systèmes expérimentés.

Face à ce problème, les concepteurs essaient de favoriser la réutilisation de code, en s'orientant vers l'assemblage de blocs préconçus et pré-vérifiés désignés sous le terme de composants virtuels ou IP (pour intellectual property). Ce concept, né dans le milieu des années 90, a donné lieu à plusieurs vocables pour désigner ces blocs réutilisables : composants réutilisables, composants virtuels, IPs (Intellectual Properties), ou plus simplement macros.

Ces dits systèmes intègrent un code de plus en plus volumineux et des parties matérielles dont la complexité est en croissance exponentielle, cela se conjugue mal avec leurs contraintes spécifiques et en particulier celles des systèmes

embarqués, qui présentent essentiellement des contraintes d'espace d'embarcation, de coût et de temps de mise sur le marché réduits (time to market) .

Ainsi, de nouvelles techniques de conception doivent-elles être trouvées. Les contraintes à prendre en compte sont typiquement les suivantes :

- Réduire au maximum le temps d'arrivée d'un produit sur le marché (time to market)
- Garantir un résultat de bonne qualité (performance, surface, consommation)
- Fiabiliser le cycle de développement (augmenter la complexité d'un circuit rend sa vérification plus difficile).

L'apparition d'un marché des composants virtuels, entend répondre au besoin des concepteurs de concentrer leurs efforts sur la partie réellement innovante d'un système plutôt que sur la co-conception de composant déjà existants [1]

pour diminuer le coût de développement d'un système sur puce, il faut :

- Connaître les bons composants virtuels à intégrer (disposer d'un large catalogue d'IP),
- Savoir comment les intégrer de manière efficace (appliquer une méthodologie de réutilisation).[7]

Le travail que nous présentons dans ce papier a permis de définir des méthodes de conception et de réutilisation afin de construire et utiliser des composants virtuels (IPs).

L'article est divisé en sections : dans la suite, nous rappelons en section 2 la signification d'un IP, la section 3 est consacrée sur la catégorisation des IPs . la section 4 définit les acteurs des IPs. Quant à la section 5 et 6, elle exhibe la méthodologie de conception et de réutilisation ainsi que les détails d'implémentation et d'évaluation sur des exemples réels. La section 7 achève ce papier par une conclusion.

## 2. Etat de l'art

Un composant virtuel est la spécification d'un composant réalisant une fonction bien déterminée pouvant être synthétisé, donc réutilisé, par un utilisateur n'ayant pas participé à la spécification de ce composant. Cette démarche comprend que seule une appréhension de la fonction réalisée est nécessaire, en aucune manière les détails de la spécification. Cependant cette absence de connaissance détaillée ne doit pas empêcher

l'obtention par l'utilisateur de performances identiques après synthèse, à celles obtenues par le créateur de l'IP.

Un tel composant correspond aussi à un ensemble de services pour [2] :

- l'évaluation de performances en vue de son intégration dans un système.
- les scripts de synthèse (logique, physique) qui garantissent à l'utilisateur de l'IP d'atteindre les mêmes performances physiques que celles obtenues par le fournisseur.
- les fichiers de test, à tous les niveaux d'abstraction, qui permettront de valider l'intégration de l'IP.

### 3. Catégories d'IP

VSIA définit trois classes de composants virtuels matériels en fonction du niveau d'abstraction de leur description synthétisable [3][9].

#### 3.1. IP hardware

La première est la catégorie hardware ou IP matériel, elle permet d'optimiser la puissance, la taille ou les performances et le plan pour une technologie spécifique. Elle englobe la netlist entière, le routage et l'optimisation pour une librairie technologique spécifique, un layout personnalisé ou une combinaison des deux. L'inconvénient est qu'il est moins flexible et portable car le processus est dépendant de la technologie (Plate-forme), par contre il présente l'avantage d'être prédictif. La protection est meilleure car on n'utilise pas le niveau RTL.

#### 3.2. IP firmware

La seconde est la catégorie Firmware (ou flexible) : elle permet d'optimiser la structure et la topologie pour les performances et la superficie à travers le placement des composants sur le plan et dans la topologie. Le coeur de type firm inclut une combinaison du RTL synthétisable, des références de la bibliothèque technologique cible, une netlist 1 de portes complète ou partielle. Le firm core n'est pas routé, ce type de macro bloc est donc un compromis entre les IPs qui sont complètement logiciels et ceux complètement matériels. Les performances et la taille sont prévisibles mais la protection de la propriété intellectuelle peut ne pas être assurée.

### 3.3. IP software

Enfin pour la famille software (ou logiciel), le composant est livré sous sa forme HDL2 synthétisable et son principal avantage est qu'il est simple d'utilisation, c'est à dire flexible. L'inconvénient majeur est qu'il ne peut être prédictif en termes de superficie, puissance et temps. Egalement on diminue la protection de la propriété intellectuelle car le code RTL source est demandé par l'intégrateur.

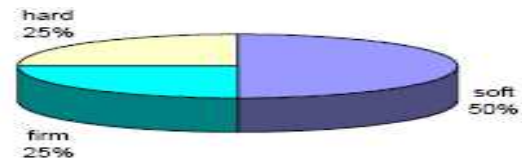


Figure1 : Répartition des IPs [8].

### 4. Les Acteurs d'IPs

Un composant virtuel fait intervenir quatre acteurs principaux [4] :

- **Le créateur ou concepteur** du composant, est l'auteur des différents fichiers délivrables qui le constituent[12].
- **L'utilisateur ou intégrateur d'un IP**, est le client final qui sélectionne les composants correspondants à ses besoins, en fait l'acquisition et l'insertion dans le système en cours de conception. Concepteur et fournisseur d'un composant s'engagent à mettre sur le marché une description fonctionnellement correcte, synthétisable, éprouvée sur au moins une cible technologique à l'aide des outils les plus répandus et munie de tous les fichiers et documents nécessaires à son implantation dans un système.
- **L'outilleur**, fournit les logiciels de CAO nécessaire à la conception, à la vérification, à la documentation, à la sélection et à l'intégration d'IP. Les outilleurs garantissent une propriété d'interopérabilité entre les flots de conception et de réutilisation. Ils s'appliquent à respecter des formats de données et des langages de spécification standards laissant à l'utilisateur le choix des outils et des cibles technologiques (ASIC, FPGA).
- **Le fournisseur**, est une entreprise ayant un lien direct ou indirect avec le créateur, qui met l'IP sur son catalogue et se charge de sa commercialisation. Les fournisseurs sont apparus face aux difficultés rencontrées par les créateurs et les concepteurs qui souhaitent faire de la réutilisation de blocs [10][11].

Les IPs, actuellement disponibles sur le marché, correspondent à des applications bien précises : **les interfaces de bus, la transmission de données, les réseaux, les microprocesseurs et microcontrôleurs** ainsi que les **périphériques**.

<sup>1</sup> Netlist : représentation du fonctionnement du composant sous forme mathématique

<sup>2</sup> HDL : Hardware Description Language

La figure 2 dresse un aperçu sur la répartition des différents types d'IPs selon les applications. Nous constatons que les éléments de mémorisation sont typiquement de type hard, les microprocesseurs et microcontrôleurs ont une répartition hard ou soft comparable, par contre les IPs de type interface bus, transmission de données et périphériques sont en grande partie en soft [5]

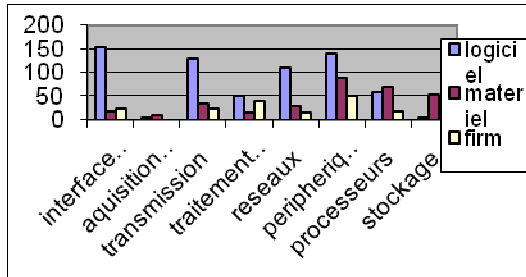


Figure2 : Pourcentage des différents d'IPs dans les applications [8].

## 5. Stratégie de Design for Reuse

Il est bien clair qu'un effort supplémentaire est nécessaire pour concevoir un module réutilisable. Les aspects de vérification et de documentation sont probablement les plus chronovores dans un projet. De plus, il faudra souvent faire quelques concessions quant au matériel nécessaire pour implémenter la fonction. Le code sera en effet souvent moins optimisé qu'un code n'implémentant que la fonction de base (pour intégrer d'autres fonctionnalités, l'architecture sera probablement moins optimale). Plusieurs stratégies peuvent être envisagées lorsqu'on envisage de concevoir un module réutilisable [2] :

Pour une stratégie **planifiée**, on présume que le bloc sera réutilisé très souvent. Ainsi, dès la conception initiale, on va faire beaucoup d'efforts pour obtenir un bloc très performant. Il va donc être tout de suite très configurable, parfaitement vérifié, pour toutes les configurations possibles, et aussi complètement documentées. Cette conception initiale demandera donc beaucoup de temps et de ressources.

On estime qu'il faudra passer environ deux fois plus de temps lors de la conception initiale (par rapport à une conception sans objectif de réutilisation). Lors de la seconde utilisation, on estime également qu'un peu de travail serait à faire. Il peut s'agir de caractéristiques "oubliées" lors de la conception initiale, ou encore d'une configuration négligée lors du test.

Par contre, les utilisations futures seront très économiques, se résumant idéalement à une lecture de documentation suivie de l'intégration du module.

Pour une stratégie **ad'hoc**, on ne sait pas encore si le module en question sera réutilisé ou non. Ainsi, on ne fera pas d'effort particulier pour le rendre réutilisable dès la conception initiale. De même, lors de chaque réutilisation, on va juste

adapter ou modifier le code existant à son nouveau contexte d'utilisation.

La conception initiale n'est donc pas plus coûteuse qu'une conception sans objectif reuse (elle est d'ailleurs en tout point identique). Par contre, chaque utilisation future restera encore assez coûteuse (50% est une moyenne, dépendant du degré de nouveauté du contexte courant d'utilisation). Même si le module sera réutilisé, il ne sera jamais réutilisable en tant que tel.

Pour une Stratégie **Redesign For Reuse**, Pour cette stratégie également, on n'est pas certain que le module devra être réutilisé souvent. Ainsi, on ne fait pas plus d'effort particulier lors de la conception initiale. Par contre, lors de la seconde utilisation et si on est désormais convaincu du potentiel reuse du bloc, on va faire des efforts de conception afin de le rendre parfaitement réutilisable. Le module devrait devenir aussi performant que le même module conçu avec une stratégie planifiée.

L'effort fourni lors de la seconde utilisation sera probablement moins soutenu que la conception initiale avec une stratégie planifiée. En effet, comme c'est une reconception, on aura une meilleure connaissance du bloc, et donc on saura mieux comment le rendre bien configurable, et la validation sera également plus facile.

On note toutefois que la conception initiale ne se fait pas sans aucun objectif de réutilisation. C'est à dire que la méthode de conception est légèrement différente d'une conception sans objectif de réutilisation. Cependant, les règles envisagées lors de cette conception initiale ne devront pas être très contraignantes. Après assimilation, leur application ne devrait pas ou très peu entraîner d'effort particulier par rapport à un bon design sans objectif particulier de réutilisation.

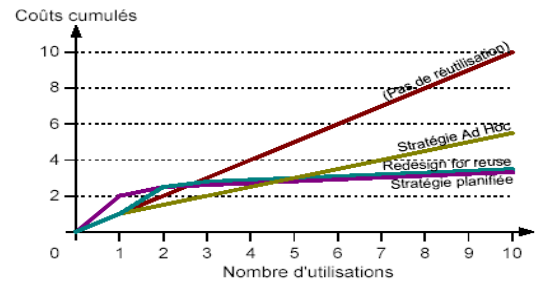


Figure 3 : Comparaison des stratégies de réutilisation

Les coûts cumulés de chaque stratégie en fonction du nombre d'utilisations montrent l'avantage d'une stratégie planifiée ou du redesign for reuse par rapport à une stratégie ad-hoc. Par contre, on voit que la stratégie planifiée est très coûteuse à la conception initiale.

La stratégie ad-hoc est presque toujours inintéressante, sauf peut-être dans le cas de blocs dont on est certain de les réutiliser très peu (typiquement moins de 4 à 5 utilisations). La figure 2 illustre les coûts associés.

Les avantages d'une stratégie de redesign for reuse sont très significatifs :

- Moins de risques de se tromper sur le potentiel reuse d'un bloc : on attend un besoin de réutiliser avant de faire des efforts sur un bloc. Il serait très gênant d'avoir fait un bloc réutilisable alors que l'on n'en a finalement pas besoin.
- Après la conception initiale, on a plus de connaissances sur le bloc, et on peut donc mieux savoir comment le rendre bien configurable et adapté à un usage général. Ici également, il serait très gênant d'avoir fait un bloc réutilisable et avoir "oublié" une ou plusieurs caractéristiques importantes qui vont le rendre beaucoup moins intéressant à réutiliser.
- L'effort de rendre le bloc réutilisable peut se faire quand on le juge nécessaire ou quand on a les ressources nécessaires (généralement à la seconde utilisation, voire plus tard dans certains cas). Le processus de reuse pourra donc commencer même sur un module se trouvant sur un chemin critique.

Pour ces raisons, c'est la stratégie qui est conseillée dans un premier temps pour mettre en place la méthodologie reuse. Par contre, une fois bien assimilée, une stratégie planifiée pourra être envisagée, lorsqu'on saura comment faire un bon bloc réutilisable.

## 6. Les Outils :

### 6.1 les outils d'aide à la conception et à la réutilisation d'IP

Les outils de conception se composent d'une part d'outils de conception système, d'autre part d'outils de conception d'ASIC<sup>3</sup>. Pour les premiers, il s'agit principalement de sélectionner et valider un IP, pour les seconds il s'agit de matérialiser un IP du niveau de spécification "soft" ou "hard" jusqu'à la définition du plan de masse. En plus, la notion d'IP a donné lieu à l'élaboration de nouvelles classes d'outils [1,4].

### 6.2 . Les outils de co-conception matériel /logiciel

Les outils d'aide à la réutilisation de composants virtuels s'inscrivent dans un temps à l'entrée de la spécification initiale sous forme mixte structurelle /comportementale, en orthogonalisant la fonctionnalité de chaque composant et les aspects liés à la communication avec son environnement. Ces outils permettent dans un deuxième temps la projection de ces modèles comportementaux et des modèles de communication sur des composants virtuels en bibliothèques et des modèles de protocole. Les principaux outils disponibles relevant de ce domaine sont :

-**VCC** (Virtual Component Co-Design) de cadence est un environnement complet de conception conjointe logiciel/matériel [10].

- **CoWare N2C** est un outil de spécification système et de conception conjointe logiciel/matériel pour les SOCs.

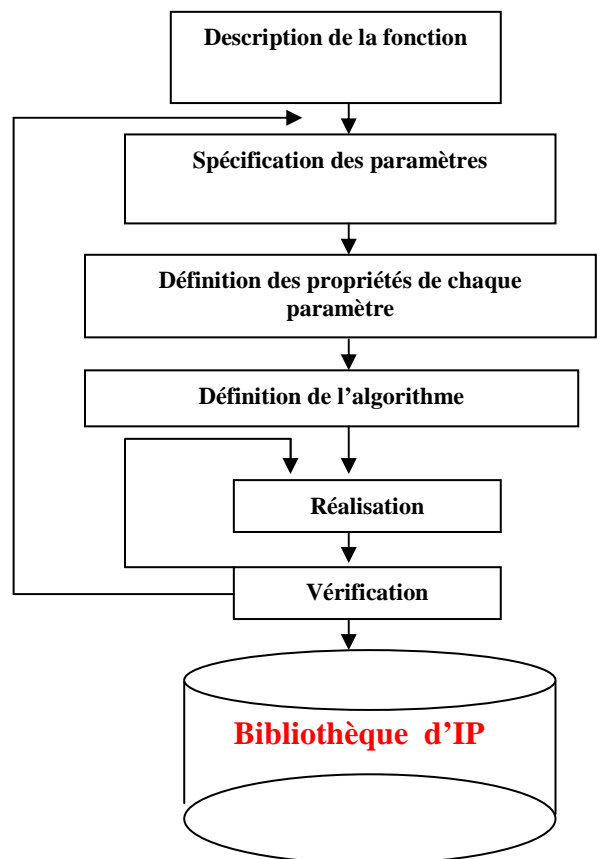
### 6.3. Outils de synthèse logique

La synthèse logique consiste à transformer une spécification logique exprimée dans un langage de programmation en une netlist logique que l'on pourra ensuite mapper vers une technologie donnée. Il existe plusieurs environnements de synthèse logique commerciaux tels que : **Synergy** et **Ambit** de **Cadence**, **Leonardo** de **Mentor Graphics**,

**Remarque :** Il existe aussi des outils de qualification de composants virtuels tels que : **QuickUse** Qualification system de **Mentor Graphics**; de même qu'il existe des outils de gestion de catalogues et de bibliothèques de composants : **IP E\_Design Manager** de **Design & Reuse** et **IP Gear** de **Synchronicity**.

## 7. Architecture du système

La figure 4 présente le flot de conception d'un IP. Nous décrivons ci-après les principales entités :



**Figure4** : Flot de conception d'un IP

- ✓ **Description de la Fonction** : dans cette étape, on décrit la fonction à réaliser dans un format de haut niveau qui peut être un graphe, un pseudo algorithme ou un langage naturel

<sup>3</sup> ASIC : Application Specific Integrated Circuit

✓ **Conception:**

- **Spécification des paramètres :** le choix du sous ensemble des paramètres est effectué selon le degré de généralité fixé au préalable. Ce dernier influe sur la complexité de l'algorithme.
- **Définition des propriétés de chaque paramètre :** les paramètres peuvent être continus ou discrets, en rapport avec l'exigence du problème.
- **Définition de l'algorithme :** l'algorithme est conçu selon la description de la fonction, de la spécification des paramètres et leurs propriétés.
- **Réalisation :** L'algorithme est implémenté avec un outil de synthèse de haut niveau : *Quartus II*, ou *VC++*, selon la nature de l'IP à réaliser.
- **Vérification :** après la réalisation, le contrôle du bon fonctionnement de l'IP est indispensable, afin de détecter des erreurs éventuelles. Les sources d'erreur peuvent être dues à la conception ou à la réalisation.

✓ **Bibliothèque d'IPs :** Une fois le processus de conception et de réalisation accompli, l'IP est intégré dans la bibliothèque d'IPs muni d'une documentation complète.

Nous décrivons les principales entités de la conception d'un système avec réutilisation :

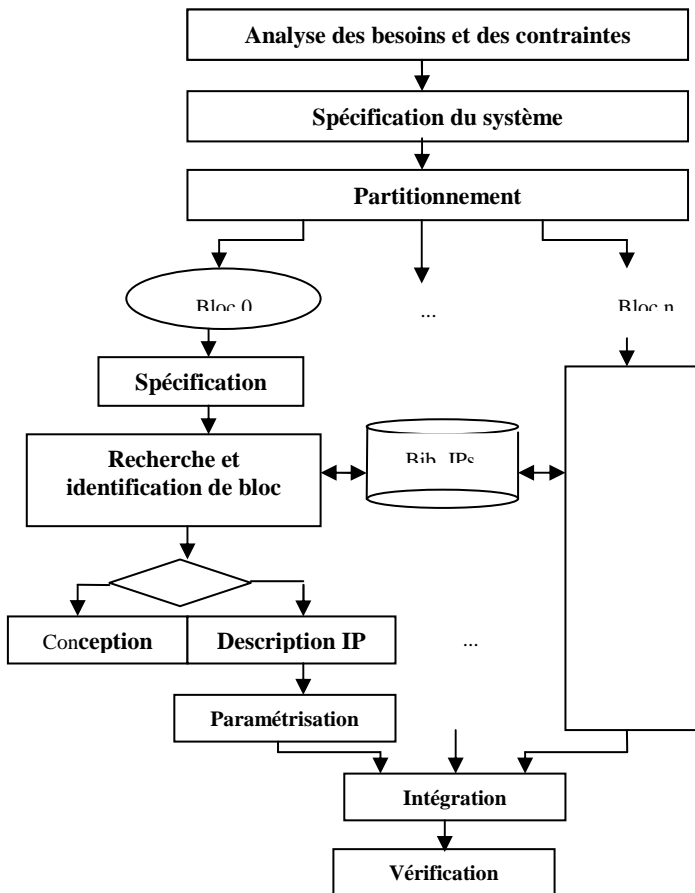


Figure5: Processus de fabrication détaillé des SOC à l'aide des IPs

**Analyse des besoins et des contraintes :** Cette étape a pour but de bien comprendre les besoins et les contraintes (exprimées dans le cahier des charges) à satisfaire dans le système conçu.

**Spécification du système :** Elle a pour but d'élaborer une description externe la plus complète possible du système à concevoir et ceci à partir de l'analyse des besoins et des contraintes.

**Partitionnement :** Elle consiste en la division du système en blocs dont la fonction de chacun est bien précise.

Après l'étape de partitionnement, l'utilisateur consulte notre bibliothèque de composants virtuels pour chacun des blocs résultant du partitionnement. S'il trouve l'IP correspondant à son bloc, il procède à son paramétrage, sinon il cherche une solution appropriée (conception du bloc).

**Intégration :** La dernière étape consiste en l'intégration des IPs paramétrés dans l'architecture globale. Pour intégrer un composant, le concepteur doit comprendre son interface, c'est-à-dire son protocole de communication avec le monde extérieur et l'insérer dans le reste du système.

**Vérification :** Cette étape permet de confirmer (ou infirmer) qu'une transformation a eu les résultats attendus. Pour avoir une base saine de vérification, il faut que les spécifications du système (et de tous ses blocs) soit la plus complète possible

## 8. Expérimentation

### 8.1 . Exemples

Le choix d'intégrer une fonction candidate ou non à la bibliothèque est effectué en respectant plusieurs facteurs, principalement l'évaluation de la réutilisation d'une fonction : c'est à dire le nombre de fois et de secteurs différents pour diverses applications dans lesquelles celle ci pourrait être réutilisée.

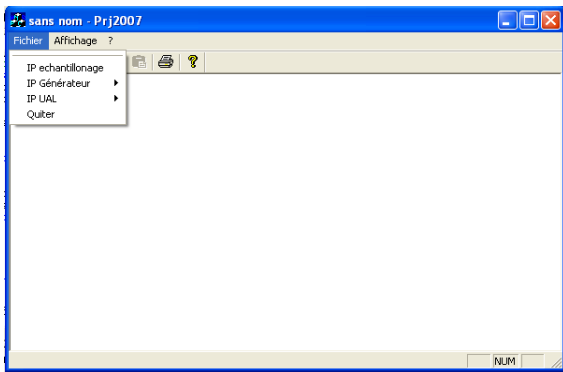
Ce facteur est tout à fait critique et stratégique pour justifier l'effort de développement et garantir la réutilisation de l'IP dans différents contextes.

Nous allons appliquer ces facteurs aux composants à concevoir un à un : Une UAL, algorithme d'échantillonnage, Générateur RSA, .

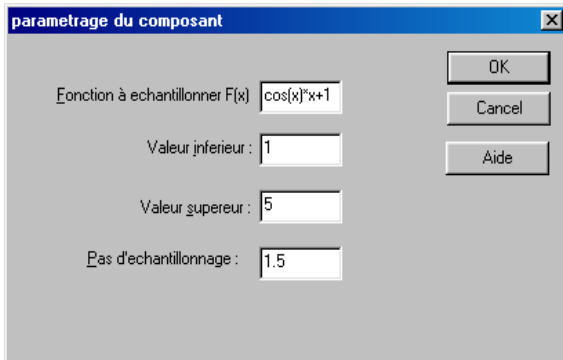
A titre illustratif, nous reproduisons seulement les écrans de l'IP échantillonnage dans la sous-section suivante.

### 8.2 Quelques écrans

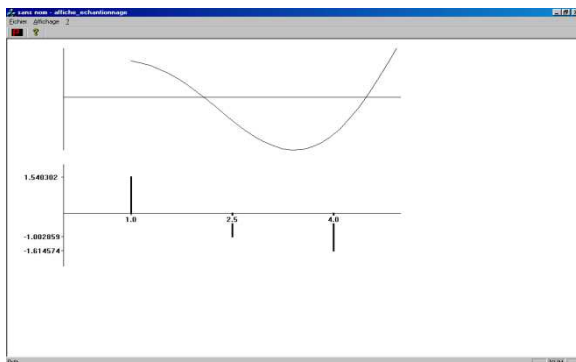
Lorsque l'application est lancée, la fenêtre suivante sera affichée figure (6) : Pour l'IP échantillonnage et pour saisir les paramètres, nous lançons la commande paramétrage du composant, une boîte de dialogue s'affiche figure (7).. Un clic sur le bouton VALIDER permet de générer un composant fonctionnant avec le paramètre choisi, la figure (8) fournit le résultat d'échantillonnage.



**Figure6** : Le menu Principal



**Figure7** : La saisie des paramètres d'échantillonnage



**Figure8** : Le résultat d'échantillonnage.

## 9. Conclusion

Notre travail s'inscrit dans une démarche de codesign où il est demandé de développer des produits de qualité, en peu de temps et à moindre coût. Le système qu'on a réalisé est une bibliothèque ouverte de composants virtuels soft permettant au concepteur de les réutiliser dans des systèmes en passant par les étapes de sélection, paramétrage et intégration. Ces blocs réutilisables (IPs) sont paramétrables donc génériques, vérifiés avec soin, portables et leurs codes sont bien commentés ce qui facilite une éventuelle modification. Ainsi, ces IPs offrent à l'utilisateur des services tels que :

- une documentation complète facilitant l'intégration de l'IP.
- des fichiers de test permettant de vérifier le bon fonctionnement de l'IP après intégration.

Nous souhaitons à l'avenir, enrichir cette bibliothèque en incluant d'autres composants virtuels, qui implémentent des fonctions plus complexes pour offrir aux clients plus d'applications.

## 10. Références

- [1] Guillaume Savaton : Méthodologie de conception de composants virtuels Comportementaux pour une chaîne de traitement du signal embarquées .
- [2] Hayet Souffî - Kebbatî : Conception d'opérateurs numériques réutilisables : Application à une méthodologie d'implantation rapide et optimale d'algorithmes de commande des systèmes électriques.
- [3] VSI Alliance, Architecture Document – Version 1.0. Rapport technique, 1997.
- [4] D.D GAJSKI et al. Essential Issues for IP Reuse . Dans Proc. Asia for and South Pacific Design Automation Conference(ASPDAC) – Embedded Tutorial , pages 37-42 , 2000
- [5] Bernard Laurent : Conception des blocs réutilisables : Réflexion sur la méthodologie
- [6] YERVANT ZORIAN, ERIK JAN MARINISSEN, and SUJIT DEY. Testing Embedded-Core Based System Chips. In Proceedings IEEE International Test Conference (ITC), pages 130.143, Washington, DC, October 1998. IEEE Computer Society Press.
- [7] ADEL BEGANNE, EMMANUEL CASSEAU, ERIC MARTIN PHILIPPE COUSSY Méthodologie pour la conception de composants virtuels IP
- [8] DESIGN AND REUSE : <http://www.design-reuse.com>
- [9] VSI Alliance, . <http://www.vsi.org>
- [10] CADENCE : <http://www.vcc.com>
- [11] XILINIX : <http://www.xilinx.com>.
- [12] ALTERA : <http://www.altera.com>