

Simulation HDL et modélisation mathématique

(06/14/2006 12:55 PM EDT)

URL: <http://eetimeseuropa.cmp.com/france/189401147>

Ce document présente une nouvelle approche dans le domaine de la simulation et de la modélisation de circuits numériques de haut niveau qui bénéficie de l'environnement mathématique délivré par MATLAB. Il permet d'intégrer le processus de conception et de vérifier directement les résultats obtenus avec des formules mathématiques ou des opérations complexes qui ne sont pas disponibles dans les langages HDL standard. Une partie du code HDL peut être placée pour des vérifications dans le modèle mathématique avancé ou peut exécuter des calculs complexes. Ce document présente les problèmes introduits par la simulation hybride et l'environnement de modélisation concernant la représentation de données, le processus de simulation et la performance optimale.

Cet article a été écrit par Adam Milik, spécialiste de la modélisation & simulation, et Adam Zytka, ingénieur d'application, chez Aldec

Résumé : Ce document présente une nouvelle approche dans le domaine de la simulation et de la modélisation de circuits numériques de haut niveau qui bénéficie de l'environnement mathématique délivré par MATLAB. Il permet d'intégrer le processus de conception et de vérifier directement les résultats obtenus avec des formules mathématiques ou des opérations complexes qui ne sont pas disponibles dans les langages HDL standard. Une partie du code HDL peut être placée pour des vérifications dans le modèle mathématique avancé ou peut exécuter des calculs complexes. Ce document présente les problèmes introduits par la simulation hybride et l'environnement de modélisation concernant la représentation de données, le processus de simulation et la performance optimale.

Introduction

Les conceptions actuelles sont basées sur le paradigme au niveau système (Ferrari et Sangiovanni-Vincentelli, 1999, Gajski 2000). Les techniques de conception au niveau système sont basées sur des composants systèmes définis avec une partie de la conception personnalisée. La partie non modifiable du système consiste généralement en un système microprocesseur avec un ensemble basique d'unités périphériques utilisées pour résoudre un problème donné (automobile, industriel, télécommunications, applications générales etc.). Les caractéristiques spécifiques du produit conçu sont placées dans la partie personnalisée du matériel et bien sûr dans le logiciel dédié. La partie personnalisée du matériel peut être implémentée comme une partie programmée masquée du composant ou comme un FPGA



2 rue Galilée
78280 GUYANCOURT
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr
info@cadvision.fr

3 place du Palais
26000 VALENCE
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr

conception au niveau système est la performance. Le problème de la performance est en grande partie dépendant du placement de la limite entre le matériel et la partie logicielle de la conception. Cette limite est contrainte par plusieurs facteurs comme la consommation en puissance, la taille de la partie FPGA, la performance du système, etc. Afin d'obtenir les meilleurs résultats, le processus de partitionnement doit être exécuté pour différentes combinaisons de solutions possibles (*Fig. 1*). La solution système présentée nécessite une nouvelle approche de développement de conception qui permet une conception et une vérification cohérente du produit durant chaque étape du développement.

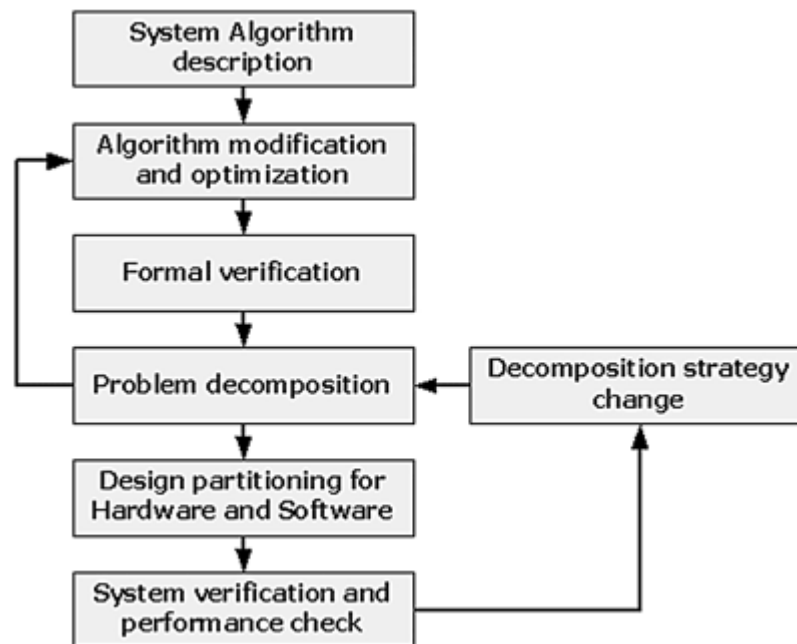


Fig. 1. Flot de conception au niveau système

Dans le passé, le concept du développement de conception était divisé en plusieurs étapes qui n'étaient pas liées les unes aux autres. Affiner la conception d'un niveau d'abstraction à un autre nécessite un effort substantiel de l'équipe de conception. Les modèles développés à un niveau d'abstraction donné peuvent difficilement être utilisés dans d'autres niveaux d'abstraction. Le développement de la conception commence à partir d'un concept modélisé avec des langages de haut niveau (C, Fortran, Pascal,...) ou des outils (Matlab, Mathcad,...). Ce niveau d'abstraction permet d'obtenir une vérification algorithmique aux toutes premières étapes du développement de la conception. La vérification dans le domaine de l'algorithmique assure seulement au concepteur que le bon algorithme a été choisi. En fait, à cette étape du développement de la conception, il est difficile d'estimer la performance du système. Une étape extrêmement importante qui suit la vérification algorithmique est le partitionnement de la conception. Un partitionnement incorrect de la conception peut mener à une implémentation inefficace qui permet difficilement d'atteindre les contraintes de la conception. D'habitude, durant le partitionnement, plusieurs facteurs sont pris en compte. En général, les algorithmes sont mieux implémentés dans le matériel que par les logiciels concurrents. L'implémentation logicielle et le processus de débogage est beaucoup plus simple et moins long que pour le matériel. Une conception correctement partitionnée doit avoir des structures matérielles flexibles qui permettent d'assurer la performance requise avec



2 rue Galilée
78280 GUYANCOURT
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr
info@cadvision.fr

3 place du Palais
26000 VALENCE
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr

il est évident qu'une approche spécifique est nécessaire pour le développement de la conception (Milik, 2005).

Outils de Modélisation Mathématique

Dans le secteur du contrôle automatique et du DSP, une approche de conception typique est basée sur des algorithmes mathématiques. Dans le domaine des algorithmes mathématiques, MATLAB et Simulink deviennent des outils très populaires (Hanselman et Littlefield, 2005). Cet environnement mathématique est approprié pour résoudre des problèmes techniques variés, depuis le traitement de signal jusqu'au contrôle du processus. Les algorithmes peuvent être examinés dans les domaines temporels discrets et variés. L'environnement est aussi équipé avec un ensemble complet d'outils d'analyse. Les blocs-diagrammes Simulink permettent de construire des modèles hiérarchiques du problème exploré. De nos jours, créer un algorithme n'est pas suffisant. La complexité de calcul est si élevée qu'un support est aussi nécessaire durant l'implémentation de l'algorithme et la vérification finale.

L'environnement MATLAB

MATLAB est un langage de haute performance pour le calcul technique. Il intègre le calcul, la visualisation et la programmation dans un environnement simple d'utilisation, où les problèmes et les solutions sont exprimés dans une notation mathématique familière. Les utilisations typiques incluent :

- Mathématique et calcul
- Développement d'algorithmes
- Acquisition de données
- Modélisation, Simulation et Prototypage
- Analyse de données, export et visualisation
- Outils graphiques scientifique et d'ingénierie
- Développement d'applications, incluant le développement d'interface utilisateur

MATLAB est un système interactif dont l'élément de données de base est un tableau qui ne nécessite pas de dimensionnement. Cela permet de résoudre de nombreux problèmes de calcul technique, plus spécialement ceux avec des formulations matricielles et vectorielles, dans une fraction de temps nécessaire à l'écriture d'un programme dans un langage scalaire non interactif tel que C ou Fortran.

Le nom MATLAB vient de matrix laboratory. MATLAB a été écrit à l'origine pour fournir un accès facile aux logiciels matriciels développés par les projets LINPACK et EISPACK. Aujourd'hui, les moteurs MATLAB intègrent les bibliothèques LAPACK et BLAS, embarquant l'état de l'art pour le calcul matriciel dans le logiciel.

MATLAB s'est développé sur quelques années à l'aide de nombreux utilisateurs. Dans les universités, c'est l'outil standard pour l'introduction et les cours avancés en mathématiques, en ingénierie et en science. Dans l'industrie, MATLAB est l'outil de choix pour la recherche de haute productivité, le développement et l'analyse.

Simulink



2 rue Galilée
78280 GUYANCOURT
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr
info@cadvision.fr

3 place du Palais
26000 VALENCE
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr

Simulink est un package logiciel pour la modélisation, la simulation et l'analyse de systèmes dynamiques. Il supporte les systèmes linéaires et non linéaires, modélisés temporellement en continu, échantillonnés, ou un mélange des deux. Les systèmes peuvent être multirate, c'est à dire avoir différentes parties échantillonnées ou mis à jour à différentes rates.

Pour la modélisation, Simulink fournit une interface graphique (GUI) pour construire des modèles en blocs -diagrammes. Avec cette interface, le modèle peut être dessiné comme il le serait avec un stylo et du papier (ou comme la plupart des livres le décrivent). On est loin des simulations qui nécessitaient de formuler des équations différentielles dans un langage ou par un programme. Simulink inclut une bibliothèque complète de blocs, de sources, de composants linéaires et non linéaires ainsi que des connecteurs. L'utilisateur peut aussi personnaliser et créer ses propres blocs en écrivant les fonctions S qui décrivent le bloc.

Les modèles sont hiérarchiques, donc les systèmes peuvent être construits avec les deux méthodologies top-up et top-down.

Après avoir défini un modèle, il peut être simulé, en utilisant un choix de la méthode d'intégration. Des scopes et d'autres blocs d'affichage peuvent permettre de visualiser les résultats de simulation tant que la simulation est en cours. De plus, plusieurs paramètres peuvent être changés afin de voir ce qui se passe pour l'exploration "what if". Les résultats de simulation peuvent être mis dans l'espace de travail de MATLAB pour la post-procession et la visualisation.

Simulation HDL avec MATLAB

Comme on l'a déjà mentionné, la complexité des systèmes numériques, qui sont conçus, nécessite des outils appropriés pour la simulation et la vérification. Pour faire basculer la conception sans à-coups entre les différents niveaux d'abstraction, l'interopérabilité des outils est nécessaire. Une telle interopérabilité établit le lien entre les différents environnements de modélisation et permet aux équipes de conception de se concentrer sur la résolution des problèmes pendant que le processus de modélisation tire avantage des différents environnements de simulation.

Un environnement typique de choix pour la conception matérielle et la modélisation utilise HDL (VHDL ou Verilog) (IEEE 1993, IEEE 1995, Ashenden 1996, Palnitkar 1996). Ces langages ne supportent pas les fonctions de modélisation mathématiques avancées. Les simulateurs conçus pour ces langages se focalisent sur la simulation la plus rapide possible du monde numérique discret. Les outils de visualisation disponibles sont dédiés aux signaux numériques. Il est presque impossible de trouver dans les waveforms des caractéristiques mathématiques des signaux. D'un autre côté, l'unité sous tests peut nécessiter un banc d'essais algorithmique spécifique qui applique aux signaux calculés des formules mathématiques. Il est très problématique de fournir de telles valeurs à l'environnement de simulation tant que plusieurs étapes sont nécessaires pour transférer des données à travers différents environnements de modélisation.

Lien vers l'environnement de simulation



2 rue Galilée
78280 GUYANCOURT
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr
info@cadvision.fr

3 place du Palais
26000 VALENCE
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr

Il est évident que la modélisation mathématique de haut niveau est nécessaire dans la simulation HDL. Le rendre possible nécessite de construire une interface qui permet de transférer des données et des commandes à travers différents programmes. Heureusement, les simulateurs MATLAB et HDL permettent d'appeler des fonctions écrites par l'utilisateur. Ces fonctions doivent respecter des contraintes données. Pour HDL, les règles d'écriture de fonctions sont décrites par le LRM approprié. Pour Verilog, le standard PLI (IEEE 1995, Sutherland 2002) est largement utilisé. Dans le cas de VHDL, il existe plusieurs interfaces. En tant que simulateur de référence, Active-HDL d'Aldec a été utilisé. Active-HDL supporte l'interface VHPI. Malheureusement, le processus de standardisation pour le VHPI est encore en chemin.

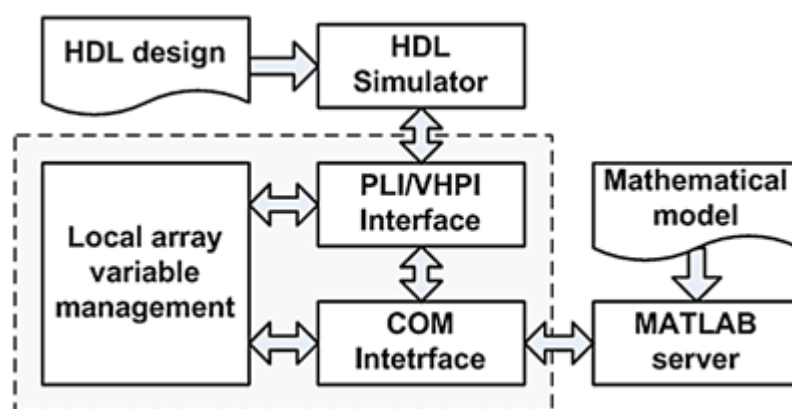


Fig. 2. Interfacer MATLAB depuis le simulateur HDL

Le concept de lier les environnements de simulation est basé sur l'utilisation du serveur de simulation MATLAB et un simulateur HDL comme client qui s'y connecte. Cette approche permet d'appeler les fonctions MATLAB durant la simulation du code HDL, comme il est nécessaire. L'interface doit supporter le transfert de données bidirectionnel et l'exécution de commandes.

Le problème de base est la communication interprocessus entre les simulateurs. MATLAB enregistre un ensemble de fonctions avec le système d'exploitation. (Windows) qui permet de l'appeler au travers l'interface COM (Common Object Model) (Rogerson, 1997). D'un autre coté, le simulateur HDL est nécessaire pour construire l'ensemble de fonctions approprié qui appelle le serveur COM de MATLAB. La fonctionnalité qui doit être supportée par l'interface est limitée aux opérations suivantes :

- Exécution de commandes
- Obtenir une variable de MATLAB
- Mettre une variable à MATLAB
- Transfert de report de Console

Une telle fonctionnalité est supportée par l'interface COM du coté MATLAB. Toutes les données qui sont transférées au travers de l'interface COM doivent être représentées dans la forme d'une variable Variant.

Conversion de Données



2 rue Gallée
78280 GUYANCOURT
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr
info@cadvision.fr

3 place du Palais
26000 VALENCE
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr

Le type variant est une structure d'information générale qui a été développé pour les besoins de l'interface COM. Cette structure permet de transférer tous les types du langage C de base. Les types Variant permettent de transférer non seulement les données scalaires mais aussi les tableaux multidimensionnels. Le type Array est une représentation de données basique pour MATLAB. Le transfert des Arrays convient parfaitement pour ce but. En général, MATLAB utilise le type point flottant pour les calculs. Contrairement à MATLAB, un système de modélisation HDL est basé sur l'utilisation des variables qui représentent des valeurs logiques. Une conversion de données correcte entre deux environnements différents est extrêmement importante.

Les valeurs scalaires (bit simple) sont traitées comme variables booléennes.

En général, chaque vecteur logique peut être considéré comme un entier ou comme une valeur à point fixe. La taille des vecteurs logiques peut être changée et normalement celle-ci n'est pas contrainte. Des différences apparaissent aussi dans la représentation de données en VHDL et Verilog. La différence la plus significative est le manque de support pour les tableaux à multi dimensions dans le Verilog IEEE 1364-1995. Pour l'interface, seulement les valeurs scalaires et vectorielles ont été sélectionnées pour représenter les variables dans les deux environnements HDL.

Les valeurs vectorielles représentent les nombres entiers ou les valeurs à point fixes. Pour la commodité de l'utilisateur, le support des deux méthodes de conversion *signed* et *unsigned* a été mis en œuvre. Chaque valeur à point flottant est convertie en un nombre fractionnel en fixant l'emplacement du point. L'emplacement du point est utilisé pour échantillonner le point flottant avant la conversion.

Comme les valeurs à point fixe ne couvrent pas l'entière plage des nombres à point flottant, il y a une plage limitée de valeurs qui peuvent être représentées comme des valeurs à point fixe. Lorsqu'une valeur convertie est hors de la plage alors la valeur du point fixe est saturée à sa valeur maximale ou minimale selon le sens du débordement.

Processus de Simulation et Performance

Dans la première approche, l'interface a été mise en œuvre avec la fonctionnalité d'exécution de commande et de transfert de valeurs scalaires. Ceci permettait de vérifier la fonctionnalité et de reconnaître l'interface comme un outil très utile pour l'extension de modélisation du système. Il permettait aussi d'observer la performance de l'interface et d'éventuels problèmes rencontrés durant le processus de simulation et l'adaptation de la conception.

Comme on l'a déjà mentionné, MATLAB exécute en général du calcul sur des matrices. Les limitations de l'interface nécessitent de convertir les données transférées en valeurs scalaires. Il y a des facteurs qui rendent cette approche inconfortable pour le concepteur et qui réduisent la performance du processus de simulation. Il est nécessaire de séparer le tableau de données en entier en valeurs scalaires avant de le transférer vers un simulateur HDL ou lorsque les données sont obtenues depuis le simulateur. Elles doivent alors être fusionnées sous la forme d'un tableau avant l'exécution du calcul dans MATLAB. Cette approche nécessite des opérations supplémentaires et plusieurs transferts entre MATLAB et le simulateur HDL. Il est



2 rue Galilée
78280 GUYANCOURT
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr
info@cadvision.fr

3 place du Palais
26000 VALENCE
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr

évident que la performance est détériorée. L'impact sur la performance est visible spécialement lorsqu'il y a un grand nombre de transferts de données interprocessus comme chaque objet du tableau est transféré séparément.

L'expérience acquise dans le processus a été utilisée pour introduire une nouvelle fonctionnalité qui est plus naturelle pour MATLAB et qui permet d'augmenter les performances de simulation. La nouvelle idée qui a été introduite est basée sur l'implémentation du gestionnaire de tableau du côté du simulateur HDL. Cette extension permet de maintenir des tableaux nommés multidimensionnels. L'utilisateur peut créer ou détruire des tableaux nommés multidimensionnels dans le système HDL. L'ensemble de fonctions PLI/VHPI permettent d'accéder aux informations relatives aux tableaux et de modifier leurs contenus. La gestion du système de tableau permet de réduire le nombre de connections interprocessus lorsque le tableau entier est transféré en une fois. Avec le nombre grandissant d'éléments tableaux, l'efficacité de la simulation est augmentée au travers de la réduction des transferts de données interprocessus. Les objets tableaux sont sélectionnés localement dans le simulateur HDL depuis une version des données mathématiques locale. De plus, le temps d'accès est réduit lorsque l'on accède aux objets par des identificateurs uniques au lieu de leurs noms.

Exemples de Simulation et Comparaison de Performance

```
$eval_string("i = 0;");  
for(i=1; i <16385; i = i + 1) begin  
    x = i;  
    $put_variable("x",x,16);  
    $eval_string("a(i) = x;");  
    $eval_string("i= i + 1;");  
end
```

Fig. 3. Opérations scalaires basiques dans un exemple HDL-MATLAB

Considérons un exemple simple de transfert de variables scalaires et l'exécution de calcul dans l'environnement MATLAB (*Fig. 3*). Les tâches suivantes sont définies pour cet exemple :

- \$eval_string – évalue une expression dans MATLAB
- \$get_variable – transfère une variable depuis MATLAB vers un environnement HDL
- \$put_variable – transfère une variable depuis le simulateur HDL vers MATLAB

L'exemple présenté permet de créer un tableau de 16384 objets. Sur la machine référence, il prend 56 secondes pour réaliser cette tâche.



2 rue Galilée
78280 GUYANCOURT
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr
info@cadvision.fr

3 place du Palais
26000 VALENCE
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr

```

$eval_string("workspace");
arr_id = $create_array("i",MAX_SIZE,1);
for(i=1; i <= MAX_SIZE; i = i + 1) begin
    $put_item(val,0,arr_id,i,1);
    val = val + 32'd1;
end
$hdl2ml(arr_id);

```

Fig. 4. Exemple d'opération Tableau

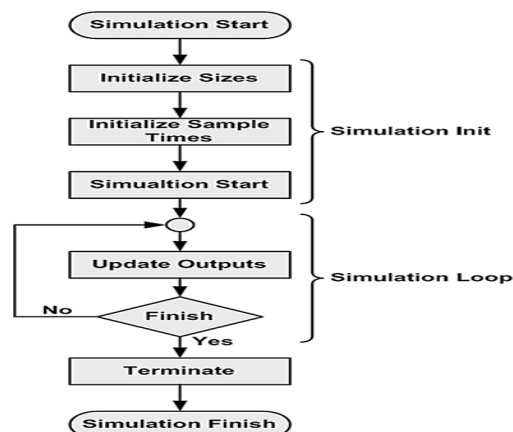
Une gestion de données plus efficace est offerte par le système de gestion de tableaux. Un exemple d'utilisation des fonctions de tableaux est présenté sur la **Fig. 4**. Les tâches suivantes permettent de gérer les tableaux :

- \$create_array – crée un tableau avec une dimension donnée
- \$destroy_array – retire un tableau
- \$get_item – prend un objet scalaire depuis un tableau
- \$put_item – met un scalaire vers un tableau
- \$hdl2ml – transfère un tableau donné depuis le simulateur HDL vers MATLAB
- \$ml2hdl – transfère un tableau donné depuis MATLAB vers un stockage de tableaux accessible par HDL

Le mécanisme de gestion de tableaux offre une meilleure performance que l'opération sur des valeurs scalaires. Sa performance est environ 600 fois plus rapide. L'exemple présenté permet de transférer des données vers 4194304 tableaux éléments. Dans ce cas, le transfert prend environ 21 secs.

Modélisation Simulink basée sur la simulation HDL

Un modèle construit dans Simulink est utilisé pour la vérification d'un algorithme. Il peut aussi être utilisé pour vérifier le modèle HDL final. Avant cela, le processus de simulation dans Simulink doit être considéré.



*Fig. 5. Processus de Simulation dans Simulink
Simulation dans Simulink*



2 rue Galilée
78280 GUYANCOURT
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr
info@cadvision.fr

3 place du Palais
26000 VALENCE
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr

Simulink permet de créer un bloc utilisateur dont la fonctionnalité est décrite par sa fonction S (Hanselman et Littlefield, 2005). Cette fonction consiste en plusieurs sous fonctions et rappels qui sont appelés par le moteur de simulation pour différentes raisons. Le diagramme général du processus de simulation montre l'ordre de l'appel des fonctions (**Fig. 5**). Le modèle HDL doit être embarqué dans le processus de simulation de Simulink selon les pré requis pour les blocs Simulink utilisateurs.

L'éditeur de diagrammes blocs dans Simulink permet de placer plus d'une instance du même bloc. Dans le moteur de simulation, tous les blocs qui sont instanciés des différents modules HDL sont servis par la même fonction S qui est appelée durant le processus de simulation. Afin de distinguer les blocs dans le modèle, chaque bloc garde ses paramètres uniques. L'ensemble des données utilisées pour configurer chaque bloc consiste en deux objets. L'un d'eux est un fichier de configuration spécifique qui stocke la description du bloc général depuis le simulateur HDL.

L'information du simulateur HDL est utilisée pour personnaliser la représentation Simulink du composant HDL. De plus, l'utilisateur peut modifier des paramètres du composant qui se répercute sur son comportement durant la simulation et la représentation graphique dans l'éditeur Simulink. Ce qui est spécialement importante pour l'utilisateur est: la méthode de conversion de données et l'intervalle d'échantillonnage du signal, la définition des paramètres génériques, l'horloge et la synchronisation des signaux ainsi que d'autres stimulateurs personnalisés. Le bloc est dessiné dans l'éditeur Simulink selon les prés requis utilisateur.



[Cliquez ici pour voir l'image](#)

Fig. 6. Blocs HDL dans le diagramme Simulink Pré-requis des co-simulation HDL

Pour exécuter le processus de co-simulation HDL, des paramètres additionnels qui ne sont pas appropriés à un bloc simple mais important pour la simulation de la conception sont nécessaires. Ces paramètres sont maintenus et configurés par un bloc de gestion de co-simulation HDL spécial qui garde des paramètres de co-simulation spécifiques (**Fig. 6**). L'aspect le plus important durant le processus de co-simulation est la relation temporelle entre le simulateur HDL et Simulink. La relation est contrôlée en fixant les unités de temps pour chaque simulateur.

Durant le processus de co-simulation, les diagnostics de la conception sont très importants, spécialement lorsque quelque chose ne fonctionne pas correctement. Le gestionnaire de co-simulation permet de basculer le processus de simulation dans le mode diagnostic interactif juste après le début de la simulation.

Préparer la conception pour la co-simulation

Avant le début de la simulation, Simulink élabore la conception et contrôle son intégrité. Durant cette phase, chaque bloc HDL dans le diagramme Simulink est appelé. Selon la demande du moteur de simulation, le bloc doit décrire le nombre d'entrées et de sorties, leur taille et type, la stratégie d'échantillonnage et l'intervalle d'échantillonnage. Les paramètres ci-dessus sont nécessaires pour le moteur de simulation pour la programmation correcte de tous les événements dans le modèle simulé.



2 rue Galilée
78280 GUYANCOURT
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr
info@cadvision.fr

3 place du Palais
26000 VALENCE
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr

```

`timescale 100ms/10ms
module test_80;
parameter \test_80/HDL_Black-Box1/w1 = 16;
parameter \test_80/HDL_Black-Box1/w2 = 32;
//Clock signals
reg CLK_2;
//ClockEnable signals
reg CE_2_2;
/*Signals declaration for module \test_80/HDL_Black-Box1 (sync_1)*/
reg [31:0] \test_80/HDL_Black-Box1/in1 ;
reg [15:0] \test_80/HDL_Black-Box1/in2 ;
wire [\test_80/HDL_Black-Box1/w1 -1:0] \test_80/HDL_Black-Box1/out1 ;
wire [\test_80/HDL_Black-Box1/w2 -1:0] \test_80/HDL_Black-Box1/out2 ;
/*Parameter value assignment*/
defparam \test_80/HDL_Black-Box1 .w1 = \test_80/HDL_Black-Box1/w1 ;
defparam \test_80/HDL_Black-Box1 .w2 = \test_80/HDL_Black-Box1/w2 ;
/*Block instance*/
sync_1 \test_80/HDL_Black-Box1 (
    .clk(CLK_2),
    .ce(CE_2_2),
    .in1(\test_80/HDL_Black-Box1/in1 ),
    .in2(\test_80/HDL_Black-Box1/in2 ),
    .out1(\test_80/HDL_Black-Box1/out1 ),
    .out2(\test_80/HDL_Black-Box1/out2 ));

//Clock signal generator for: CLK_2
initial begin
    CLK_2 = 1'b0;
    forever begin
        #10 CLK_2 = 1'b1;
        #10 CLK_2 = 1'b0;
    end
end
//ClockEnable signal generator for: CE_2_2
initial begin
    forever begin
        CE_2_2 = 1'b1;
        @(negedge CLK_2);
        CE_2_2 = 1'b0;
        repeat(1)
            @(negedge CLK_2);
    end
end
end
/*Simulink <-> HDL simulator communication*/
initial
    $link;
endmodule

```

Fig. 7. Listing du module Verilog généré pour la co-simulation pour le modèle dans la Fig. 6

La phase d'élaboration dans Simulink est utilisée pour collecter des informations et préparer le fichier top-level HDL de simulation approprié (**Fig. 7**). Selon les données collectées durant la phase d'élaboration, il est possible de préparer un fichier HDL qui contient :

- Des instances de tous les blocs
- Des assignations de valeurs aux génériques et paramètres
- Les signaux et les probes
- Des générateurs d'horloge et des signaux de synchronisation
- La fonction PLI/VHPI responsable de la communication avec Simulink

Lorsque Simulink appelle la fonction qui démarre la simulation (**Fig. 5**), le simulateur HDL est invoqué et la description HDL top-level générée est chargée pour la simulation dans le processus batch.

Il existe un problème similaire à celui qui a été décrit dans la section interface MATLAB. Il y a deux environnements de simulation qui doivent être capables de communiquer. Le niveau de complexité augmente parce que Simulink requiert une synchronisation temporelle et des valeurs de signaux, contrairement au calcul sans référence de temps dans MATLAB. Le serveur de communication



2 rue Gallée
78280 GUYANCOURT
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr
info@cadvision.fr

3 place du Palais
26000 VALENCE
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr

interprocessus est placé dans le simulateur HDL. En utilisant l'interface générique PLI/VHPI, il est possible de créer une librairie qui peut être intégrée dans le code HDL et offrir une communication interprocessus en utilisant le protocole TCP/IP ou des opérations pipe nommées (Rogerson, 1997). Ce serveur doit être capable de répondre aux demandes depuis le simulateur Simulink qui contrôle les valeurs de signaux et le temps de simulation.

Lorsqu'une simulation HDL est initialisée, Simulink fait correspondre les signaux enregistrés avec la conception HDL. Le processus de simulation normal peut démarrer juste après que l'intégration des simulateurs s'est réalisé.

Processus de simulation et sa performance

Le simulateur Simulink gère le processus de simulation tant que le processus de simulation travaille comme serveur de simulation (**Fig. 8**). Afin d'exécuter efficacement la co-simulation, le nombre de transferts de données d'interprocessus doit être limité. Cette observation contredit l'idée de simulation basée sur l'échange fréquent de petites parties de données. Ce facteur influence grandement la performance de la simulation. Ainsi, dans l'interface, une optimisation a été introduite pour collecter le plus grand nombre de données ou de requêtes.

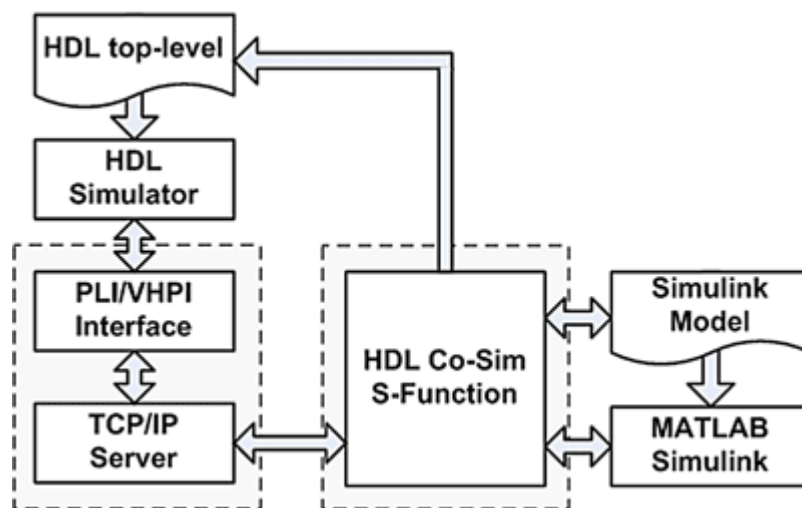


Fig. 8. Co-simulation HDL dans Simulink

Dans la fonction S, la liste des requêtes vers le simulateur HDL est intégrée. Toutes les requêtes qui doivent être exécutées sont stockées là. Lorsque le processus de simulation éteint le point qui requiert une réponse depuis le simulateur HDL, la liste de requêtes est transférée et une réponse du simulateur HDL est attendue. La réponse délivrée a aussi une forme de reconnaissance de requête. Il consiste en demandes avec l'état d'exécution et les données retrouvées. L'approche décrite permet de réduire le nombre de transferts entre les simulateurs et d'augmenter la performance en simulation.

Conclusion

La recherche montre une possibilité de connecter un environnement de modélisation mathématique populaire à des simulateurs HDL. Toutes les tâches ont été effectuées en utilisant le simulateur Active-HDL d'Aldec. Ses avantages incluent la possibilité de valider l'accès aux modèles HDL dans l'environnement Simulink et aussi en utilisant le calcul mathématique et la visualisation avancée dans le processus de simulation HDL. Une simple méthodologie de communication d'interprocessus a été montrée. La communication interprocessus utilise des mécanismes implémentés dans l'O.S. qui n'est pas optimisé pour les simulations. Des optimisations ont été introduites pour dépasser les



2 rue Gallée
78280 GUYANCOURT
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr
info@cadvision.fr

3 place du Palais
26000 VALENCE
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr

représentation de données liées est aussi extrêmement importante. Les deux environnements décrits utilisent différentes méthodes de représentation de données. Les fonctions de conversions implémentées sont flexibles et permettent d'obtenir une large plage de représentations dans le domaine HDL.

Il y a encore beaucoup de problèmes qui requièrent de plus grandes investigations. La performance du système est un des facteurs clés. Dans cette partie, il y a une possibilité d'approfondir les optimisations basées sur une meilleure compréhension du processus de simulation et des structures de données dans Simulink. Il est aussi possible d'introduire la communication à distance des simulateurs en utilisant le protocole TCP/IP. Enfin, il est possible de placer le processus de conception entier dans l'environnement Simulink.

REFERENCE

- . Ashenden P.J. "The Designer's Guide to VHDL", Morgan Kaufman Publishers, San Francisco 1996
- . Hanselman D.C., B. Littlefield "Mastering MATLAB 7", Prentice Hall, New York 2005
- . IEEE "IEEE Standard VHDL Language Reference Manual", IEEE 1993
- . IEEE, "IEEE Standard Hardware Description Language Based on the Verilog® Hardware Description Language", IEEE 1995
- . Palnitkar S. "Verilog HDL" SunSoft Press 1996
- . Rogerson D. "Inside COM" Microsoft Press 1997
- . Sutherland D., "The Verilog PLI Handbook: A Tutorial and Reference Manual on the Verilog Programming Language Interface" Kluwer Academic Publishers, 2002
- . Ferrari A., A. Sangiovanni-Vincentelli, System Design. "Traditional Concepts and New Paradigms". Proceedings of the 1999 Int. Conf. On Comp. Des, Oct 1999, Austin
- . Gajski, D., SpecC: Specification Language and Methodology. Kluwer Academic Publishers, Norwell MA, 2000
- . Milik A. "System level design in System-C using transaction-level modeling", Embedded Systems, March 2005



2 rue Galilée
78280 GUYANCOURT
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr
info@cadvision.fr

3 place du Palais
26000 VALENCE
Tel : 01.39.30.65.06
Fax : 01.39.30.65.08
www.cadvision.fr