

VÉRIFICATION

# Les assertions de SystemVerilog allègent la vérification des SoC

*Les assertions apportent des mécanismes de contrôle de comportements non désirés d'une conception et procurent une meilleure observabilité du code testé.*

*En parallèle, les débogueurs facilitent l'adoption des méthodologies de vérification à base d'assertions, grâce au support des langages d'assertions, au suivi automatique de trace...*

Classiquement, la conception et la réalisation des systèmes sur puce (SoC) complexes sont facilitées par des stratégies d'acquisition de blocs de propriété intellectuelle issus de tierce partie, par l'approche de type « diviser et conquérir » au sein des équipes de développement et, bien entendu, par l'emploi de concepteurs supplémentaires. Mais la vérification de ces grands projets implique de les aborder comme un tout. Cette lourde tâche incombe aux outils de vérification et aux méthodologies associées qui doivent autoriser la simulation d'un projet à de multiples niveaux d'abstraction. Le mouvement vers la vérification à base d'assertions vise à alléger cette tâche en améliorant dans l'environnement de vérification, d'une part, la notion d'« observabilité » c'est-à-dire de la vérification des résultats et, d'autre part, le test lui-même en facilitant le développement de procédures adaptées.

Rappelons que les assertions sont des éléments de code qui expriment un comportement désiré, ou non désiré du design. Elles donnent une description concise de la spécification de conception indépendamment de l'implantation de niveau RTL. Elles peu-

Par Bindesh Patel,

**SpringSoft**

Directeur de marketing technique chez SpringSoft et responsable de la définition des futurs produits au sein de la société, Bindesh Patel est diplômé en ingénierie informatique de l'université de Californie de Santa Cruz. Il a précédemment occupé des postes d'ingé-



nier de conception et d'application chez LSI Logic, Zycad et Atrenta.

vent être codées à l'aide de langages traditionnels comme Verilog, VHDL ou C, mais ces langages traditionnels génèrent un grand nombre de lignes de code et impliquent des techniques de programmation lourdes. A la

différence des langages spécialisés comme SystemVerilog Assertions (SVA) ou PSL (Property specification language) qui permettent de donner un énoncé plus concis des assertions (figure 1).

Prenons un exemple : le cas du couple d'instructions « fork » et « join », utilisés dans le langage Verilog pour modéliser la nature temporelle d'un comportement classique (« Après l'établissement d'un signal de requête, le signal d'acquiescement est établi de 1 à 3 cycles plus tard. »). Pour vérifier ce comportement, le langage SVA fournit une syntaxe spécialisée qui décrit ce point de manière efficace, avec notamment le symbole ## qui signifie « un nombre donné de cycles plus tard », ce nombre de cycles étant représenté par un nombre absolu ou par une fourchette. D'autres variations plus complexes sont également possibles avec SVA, alors que le code Verilog équivalent serait presque im-

## Utilisation d'un langage d'assertion

FIGURE 1

Les langages d'assertion décrivent avec concision un comportement qui peut être vérifié. Dans cet exemple, on voit la différence entre le langage Verilog et le langage SVA pour exprimer le comportement suivant : « Après l'établissement d'un signal de requête, le signal d'acquiescement est établi de 1 à 3 cycles plus tard. ».

