TP4 : Débogage du circuit Am2901

A) Objectifs

Le but de cette séance est de vous apprendre double:

- Vous apprendre, par la pratique, les méthodes de débogage.
- Vous inciter à analyser en détail les fonctionnalités du circuit Am2901 qui sera réutilisé dans les TP suivants.

B) Méthode de débogage

On se place dans une contexte où on utilise une technique de simulation pour analyser le comportement du modèle, et détecter d'éventuelles erreurs dans ce modèle. La méthode de débogage comporte 5 grandes étapes:

1. Créer une situation de dysfonctionnement

Cette étape peut être effectuée de deux manières (en général, elles sont utilisées conjointement). On peut attendre qu'un utilisateur du modèle signale un bogue. Avec cette méthode, on s'expose à des manifestations de mauvaise humeur de la part de l'utilisateur mécontent. On cherche donc à prévenir les problèmes, en développant différents scénarios (ou *jeux de test*). Le but est alors de maximiser le *taux de couverture*, en activant le maximum de fonctionnalités du circuit ou du système analysé.

2. Identifier un comportement anormal

La simulation d'un système soumis à une séquence de stimuli définit une **séquence d'événements ordonnés par dates croissantes**. Dans un système complexe, il n'est pas possible de vérifier toutes les valeurs de tous les signaux à tous les cycles, et on se contente d'observer certains signaux à certains instants. Par exemple, dans un système intégré sur puce contenant un coeur de processeur exécutant du code binaire stocké dans une mémoire embarquée sur la puce, on commencera par observer les valeurs affichées par le processeur sur un terminal. Dans ces conditions, il peut y avoir un retard important (plusieurs dizaines, voire plusieurs milliers de cycles) entre le moment où apparaissent les premiers symptômes observables du dysfonctionnement (affichage d'une valeur erronée), et le moment ou le système a commencé à dévier de son comportement *normal*. Il faut commencer par vérifier que le dysfonctionnement est reproductible, et qu'en relançant la simulation, on re-observe le comportement anormal.

3. Remonter à la cause du dysfonctionnement

C'est la partie la plus délicate. On sait qu'à une certaine date T1 (en pratique à un certain cycle), le système est dans un état anormal, mais on ne sait généralement pas à partir de quel cycle le système a commencé à *dérailler*. Il faut donc *remonter le temps* pour déterminer la date précise TC à partir de laquelle le système à quitté son fonctionnement normal pour entrer dans un état anormal. Pour cela, il faut commencer par retrouver un état normal du système, à une date T0, la plus tardive possible, mais évidemment antérieure à T1. Quand on a déterminé les deux dates T0 et T1 Il faut

analyser - cycle par cycle- le comportement du système entre T0 et T1 pour déterminer l'instant exact le système passe d'un état normal au cycle TC, à un état anormal au cycle TC+1. Rappelons que l'état d'un système synchrone est défini par l'ensemble des valeurs stockées dans les registres internes.

4. Analyser la cause du dysfonctionnement

Lorsqu'on a déterminer l'instant TC, il est généralement assez facile de détarminer la cause du dysfonctionnement du matériel, en analysant les fonctions de transition des différents automates impliqués dans le fonctionnement du système.

5. Corriger le bogue

Il faut prendre le temps de réfléchir sur la bonne façon de corriger le bogue, car il arrive souvent qu'une mauvaise correction d'un bogue introduise un autre dysfonctionnement. Il faut donc relancer la simulation sur l'ensemble des scénarios définis à l'étape 1, et pas seulement le scénario qui a mis en évidence le bogue corrigé. C'est ce qu'on appelle un test de *non régression*.

C) Circuit Am2901

Vous devez, au cours de cette séance, analyser et corriger les deux modèles comportementaux qui vous sont fournis. (deux fichiers .vbe par binôme). Dans chaque modèle, un bug a été délibérément introduit, et le travail consiste donc à localiser, puis à corriger ce bug. Pour vous aider dans ce travail, vous trouverez la DATA SHEET "officielle" du circuit Am2901 en bas de la page.

La validation devra être réalisée par simulation sous Asimut, à l'aide de stimuli générés avec Genpat. Pour obtenir la description d'un AMD Am2901, tapez la commande suivante :
/users/enseig/trncomun/TP/2006/TP1/Fichiers/Fournis/get2901.Linux > amd.vbe

Remarque : Le modèle comportemental du circuit Am2901 que vous allez récupérer présente quelques différences par rapport à la spécification définie par la DATA SHEET, mais ces différences ne constituent pas des bugs :

- La RAM de 16 mots de 4 bits est modélisée par un banc de registres à triple accès (un accès en écriture, deux accés en lecture). Les écritures se font sur front montant du signal d'horloge ck.
- Les drapeaux np et ng sortent la valeur '1' pour les opérations logiques.

D) Compte-Rendu

Il vous est demandé un rapport au format .pdf. Vous joindrez tous vos tests en annexe. Votre méthodologie sera évaluée en priorité sur les résultats. Par exemple, un groupe ayant identifié le bogue d'un seul des deux AMD Am2901 mais de manière intelligible recevra bien plus de points qu'un groupe ayant trouvé les bogues au hasard sans la moindre méthode.