

L'architecture SPSMALL :

Le but de cette partie est de comparer les résultats de tests obtenues sur les descriptions générées par le programme par rapport à celles qui sont décrites dans [CEFX06a & CEFX06c]. Tout d'abord, on rappelle ci-dessous la représentation abstraite de l'architecture SPSMALL (SPSMALL-BLUEB-LSV1), sous forme d'un graphe fonctionnel et temporel abstrait correspondant à une implémentation de l'architecture SPSMALL.

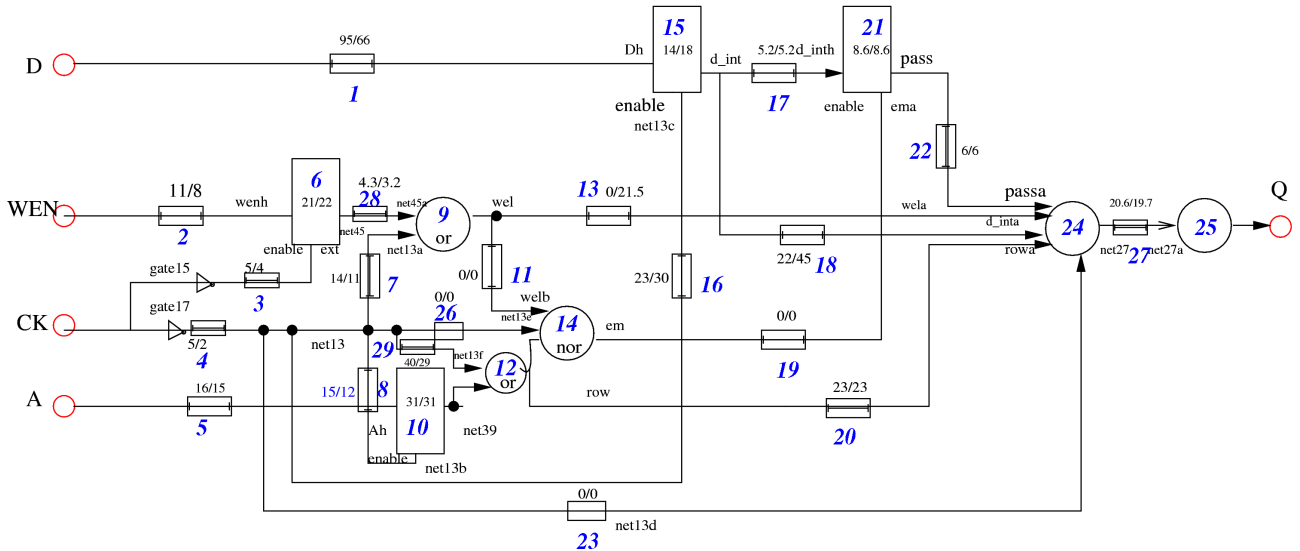


Figure 5: AFTG de SPSMALL (SPI) ([CEFX06a]).

Ici, comme a été déjà fait dans [CEFX06a & CEFX06c], on va faire l'analyse des deux implémentations SP1 et SP2 de l'architecture SPSMALL. On note que ces deux dernières sont identiques au niveau de leur représentation structurelle (leur fonction est identique). En revanche, les délais de propagation de leurs composants peuvent être différents. Ils sont représentés par des graphes fonctionnels et temporels abstraits (AFTG). Comme conséquence, les descriptions en uppaal ou hytech associées à ces deux implémentations sont presque les mêmes. Elles ne diffèrent que dans les valeurs à affecter aux paramètres associées aux délais des signaux internes. La figure mentionnée ci-dessus représente l'implémentation SP1.

Les descriptions en hytech ou en uppaal, associées à ces implémentations SP1 et SP2, générées par la programme contiennent :

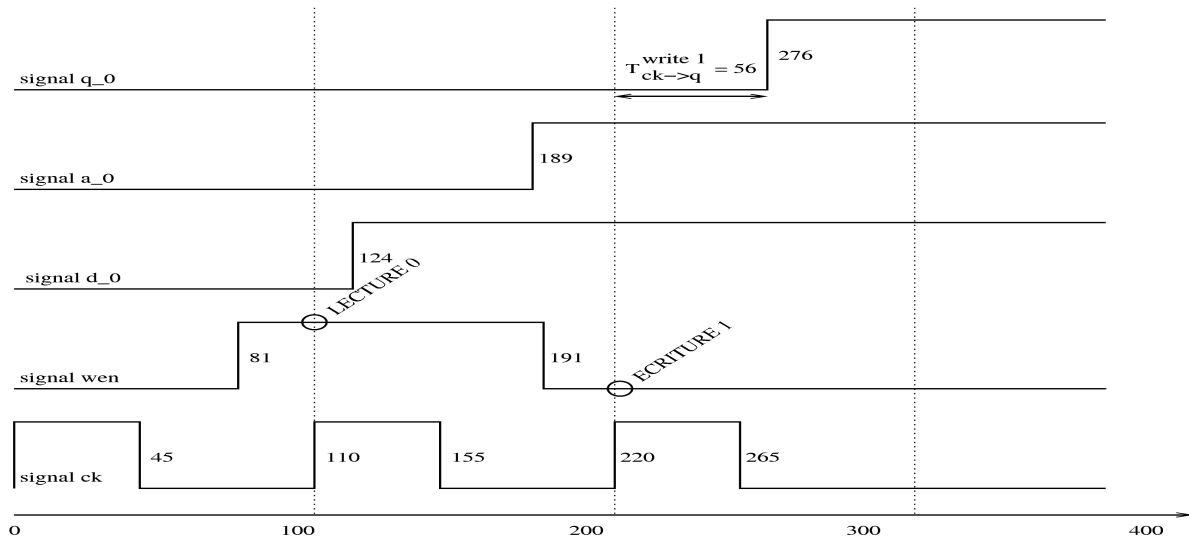
- 1553 lignes dans la description en hytech et 1291 lignes dans la description en uppaal.
- 27 automates + les automates d'environnement.
- 28 horloges.
- 32 variables discrètes.
- 62 paramètres.

Evidemment, les descriptions en hytech générées qui contiennent beaucoup d'automates ne passent pas sur l'outil Hytech. En revanche, l'outil Uppaal supporte l'analyse des descriptions de cette taille (Ceci est dû au fait déterminisme des automates). En effet, on les avait testé avec l'outil uppaal jusqu'à présent sur trois environnements différents avec lesquels on peut générer des écritures des valeurs 0 et 1 sur la sortie de l'architecture q_0.

Les résultats d'analyse sur ces trois environnements de ces classes sont décrits brièvement sur les chronogrammes suivants. Comme on voit la dessous, on a représenté que les signaux d'entrée ck, wen, d_0, a_0 et le signal de sortie q_0, pour des raisons de lisibilité.

Environnement 1:

- Avec les délais de SP1:



On note que le graphe d'atteignabilité (GA), associé à l'automate obtenu par la composition du modèle associé au programme vhdl et l'automate d'environnement satisfait la propriété suivante :

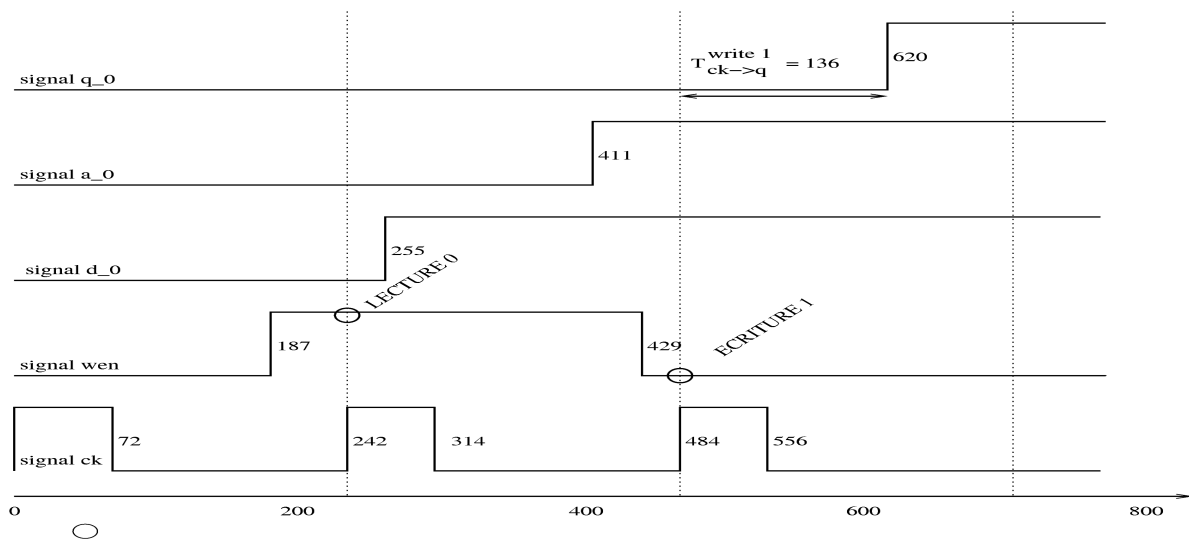
$$GA \models AG ((t > 156 \text{ and } t < 276) \text{ imply } q_0 == 0) \text{ and } (t > 276 \text{ imply } q_0 == 1))$$

Bien qu'il ne satisfait pas la propriété suivante :

$$AG ((t > 156 \text{ and } t < 277) \text{ imply } q_0 == 0) \text{ and } (t > 277 \text{ imply } q_0 == 1))$$

Cela correspond bien à ce qui est décrit dans le diagramme.

- Avec les délais de SP2:



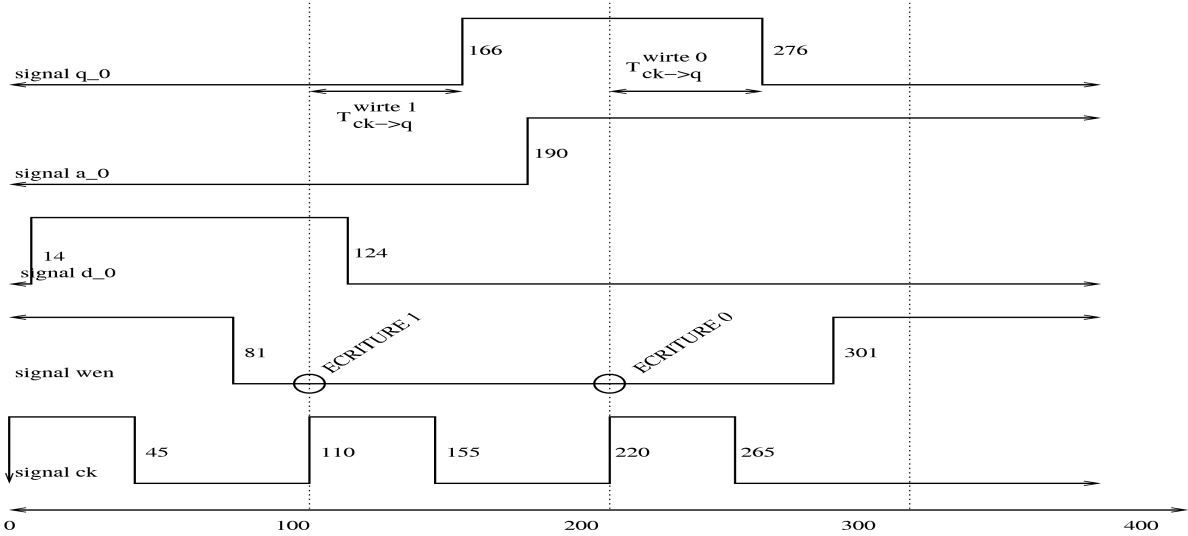
Ici, le graphe d'atteignabilité vérifie :

$$GA \models AG ((t > 383 \text{ and } t < 620) \text{ imply } q_0 == 0) \text{ and } (t > 620 \text{ imply } q_0 == 1))$$

$$GA \not\models AG ((t > 383 \text{ and } t < 621) \text{ imply } q_0 == 0) \text{ and } (t > 621 \text{ imply } q_0 == 1))$$

Environnement 2:

- Avec les délais de SPI:

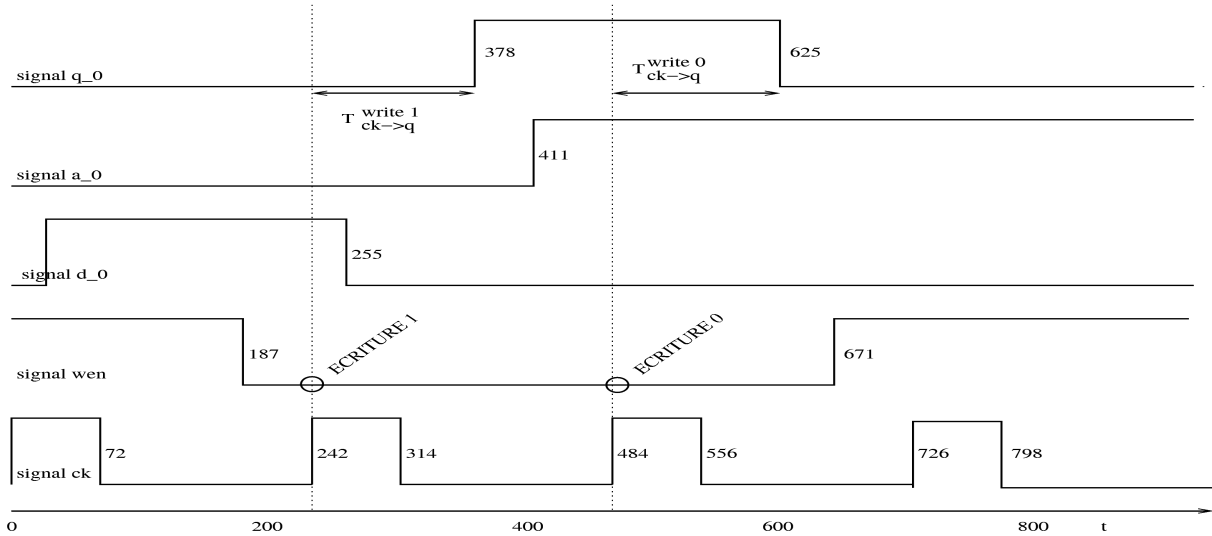


avec le graphe d'atteignabilité obtenu on a :

$$GA \models AG ((t > 275) \text{ imply } q_0 == 0) \text{ and } ((t > 166 \text{ and } t < 275) \text{ imply } q_0 == 1))$$

$$GA \not\models AG ((t > 276) \text{ imply } q_0 == 0) \text{ and } ((t > 166 \text{ and } t < 276) \text{ imply } q_0 == 1))$$

- Avec les délais de SP2:



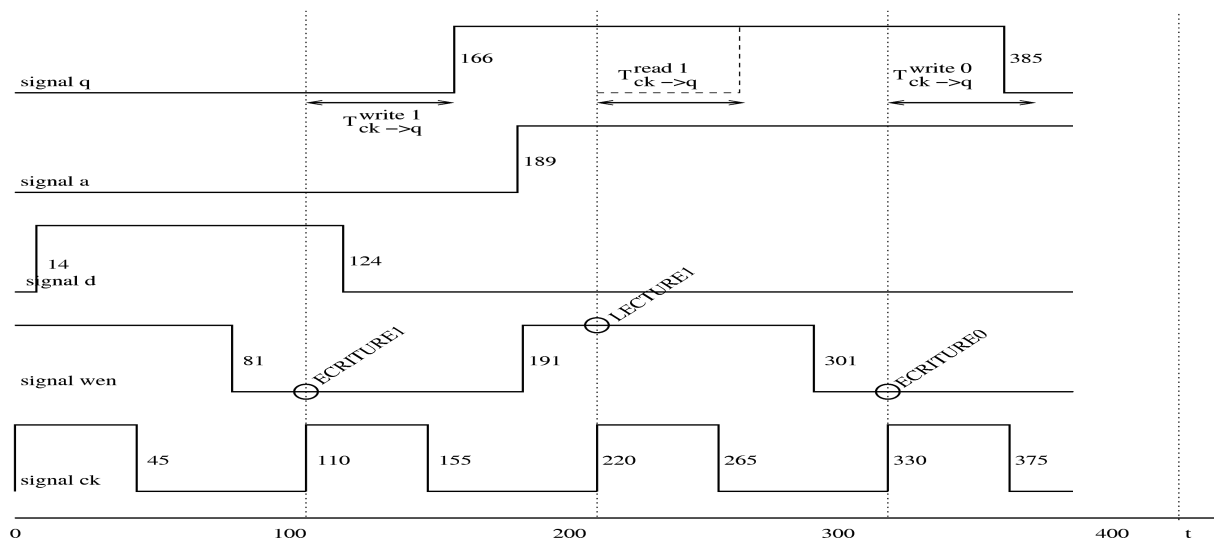
Le graphe d'atteignabilité obtenu vérifie ces propriétés :

$$GA \models AG ((t > 625) \text{ imply } q_0 == 0) \text{ and } ((t > 378 \text{ and } t < 625) \text{ imply } q_0 == 1))$$

$$GA \not\models AG ((t > 626) \text{ imply } q_0 == 0) \text{ and } ((t > 378 \text{ and } t < 626) \text{ imply } q_0 == 1))$$

Environnement 3:

- Avec les délais de SPI:

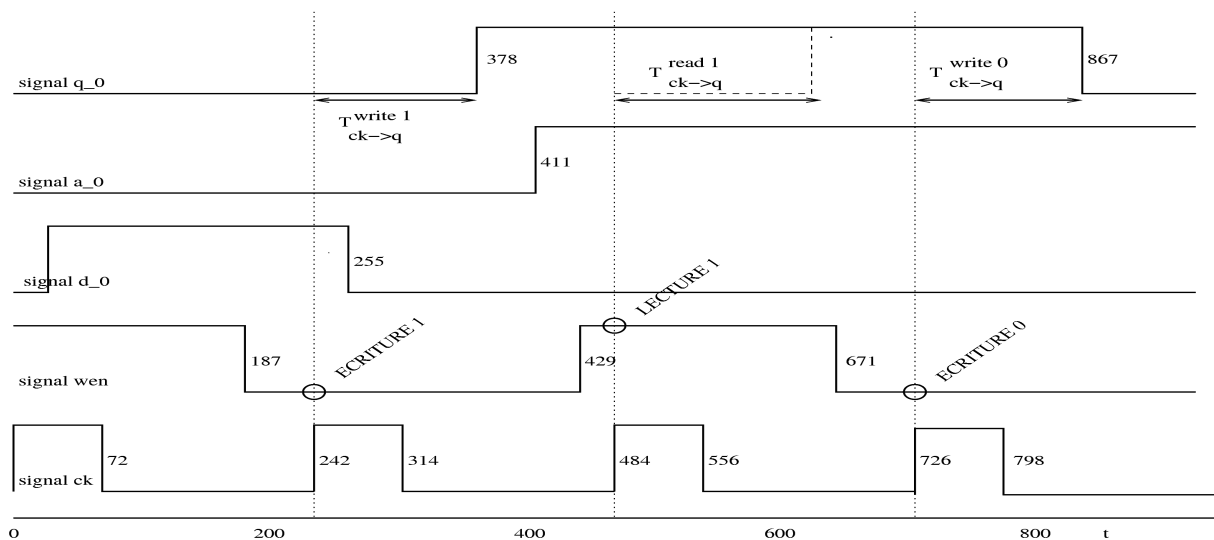


Le graphe d'atteignabilité obtenu GA vérifie que :

$$GA \models AG ((t > 385) \text{ imply } q_0 == 0) \text{ and } ((t > 166 \text{ and } t < 385) \text{ imply } q_0 == 1))$$

$$GA \not\models AG ((t > 386) \text{ imply } q_0 == 0) \text{ and } ((t > 166 \text{ and } t < 386) \text{ imply } q_0 == 1))$$

- Avec les délais de SP2:



Le graphe d'atteignabilité obtenu vérifie la contrainte suivante :

$$AG ((t > 867) \text{ imply } q_0 == 0) \text{ and } ((t > 378 \text{ and } t < 867) \text{ imply } q_0 == 1))$$

mais pas la contrainte :

$$AG ((t > 868) \text{ imply } q_0 == 0) \text{ and } ((t > 378 \text{ and } t < 868) \text{ imply } q_0 == 1))$$

On observe que les résultats des tests sont bien conformes à ce qui était attendu. En ce qui concerne l'implémentation SP1 (l'implémentation SP2), tous les temps de réponse de l'opération d'écriture des valeurs 0 et 1, notée $t_write(0)$ et $t_write(1)$, sont égaux aux valeurs 56 (136) et 55 (141) et vérifient la propriété du temps de réponse $t_write \leq t_write_max = 56$ (141), où t_write_max dénote le temps maximum nécessaire pour l'opération d'écriture sur la sortie q. Comme on peut observer sur les chronogrammes, les contraintes ci-dessus sont vérifiées pour les valeurs optimales des temps setup $t_setupd = 96$ (229), $t_setupwen = 29$ (55) ,

$t_{setup} = 31$ (73), associées aux signaux d'entrée D, WEN et A.

Les résultats obtenus et leur comparaison avec ceux cités dans [CEFX06a] sont récapitulés dans les tables mentionnées ci-dessous. On note que l'unité de temps est de 10 ps.

Pour la l'implémentation SP1:

computed response time	value of the datasheet
$t_{CK \rightarrow Q}^{read} = 74$	$t_{max}^{read} == 77$
$t_{CK \rightarrow Q}^{write} = 56$	$t_{max}^{write} == 56$

Table 1.a : Temps de réponse pour SP1, obtenus dans [CEFX06a].

computed response time
$t_{CK \rightarrow Q}^{write}(0) = 55$
$t_{CK \rightarrow Q}^{write}(1) = 56$

Table 1.b : Temps de réponse pour SP1, obtenus par notre outil.

setup parameter	optimal value obtained by computation	optimal value obtained by simulation	value of the datasheet
t_{setup}^D	95	95	108
t_{setup}^{WEN}	29	36	48
t_{setup}^A	31	30	58

Table 1.c : Temps de setup optimal pour SP1, obtenus dans [CEFX06a].

setup parameter	optimal value obtained by the programme
t_{setup}^D	96
t_{setup}^{WEN}	29
t_{setup}^A	31

Table 1.d: Temps de réponse pour SP1, obtenus par notre outil.

Pour la l'implémentation SP2:

computed response time	value of the datasheet
$t_{CK \rightarrow Q}^{read} = 169$	$t_{max}^{read} == 169$
$t_{CK \rightarrow Q}^{write} = 142$	$t_{max}^{write} == 142$

Table 2.a : Temps de réponse pour SP2, obtenus dans [CEFX06a].

computed response time
$t_{CK \rightarrow Q}^{write}(0) = 141$
$t_{CK \rightarrow Q}^{write}(1) = 136$

Table 2.b : Temps de réponse pour SP2, obtenus par notre outil.

setup parameter	optimal value obtained by computation	optimal value obtained by simulation	value of the datasheet
t_{setup}^D	229	229	241
t_{setup}^{WEN}	55	55	109
t_{setup}^A	73	74	110

Table 2.c : Temps de setup optimal pour SP2, obtenus dans [CEFX06a].

setup parameter	optimal value obtained by the programme
t_{setup}^D	229
t_{setup}^{WEN}	55
t_{setup}^A	73

Table 2.d: Temps de réponse pour SP2, obtenus par notre outil.

Il reste encore à tester les environnements qui provoquent des lectures de 0 et 1 sur les points mémoires avec d'autres représentations abstraites d'architectures SPSMALL. Le modèle SPSMALL-BLUEB-LSV1 associé à l'architecture SPSMALL ne permet pas de bien voir ces lectures car il n'a qu'un seul point mémoire pour effectuer les lectures et les écritures. On a donc besoin d'étendre l'architecture pour intégrer au moins deux points mémoires. La deuxième partie de ce document fait

l'objet de tests sur cette dernière extension.

Bibliographie

[CEFX06a]. R. Chevallier, E. Encrenaz, L. Fribourg, W. Xu, *Timing Analysis of an Embedded Memory: SPSMALL*, WSEAS 10th international conference on circuits, july 2006, Greece.

[CEFX06c]. R. Chevallier, E. Encrenaz, L. Fribourg, W. Xu, *Dynamic Timing Analysis of an Embedded Memory: SPSMALL*, Rapport interne du projet Blueberries, Juin 2006.