

 **Retour d'expérience sur outils DSP to VHDL à partir de Matlab/Simulink**

TASFR00610294-



## *Ordre du jour*

- Évolution technologique.
- Contexte et environnement d'utilisation.
- Nature des projets étudiés.
- Processus de développement utilisé
- Exemple de design
- Évaluation et comparaison
- Conclusions



### *Évolution technologique*

- Forte évolution des capacités d'intégration et de vitesse des FPGA à la fin des années 90 (famille APEX, VIRTEX...)
- Possibilité de remplacer certains ASIC's numériques par des FPGA qui par leur capacité de programmation permettent de minimiser fortement les étapes de simulations par des essais grandeur nature.
- Accélération du processus de design et de validation permettant une réduction des coûts.
- Apparition au début des années 2000, des premiers outils de transcodage VHDL (DSPBUILDER)



### *Contexte et environnement d'utilisation*

- Utilisation de l'outil dans le cadre de PEA , de démonstrateur ou de maquette mettant en œuvre des FPGA comprenant des algorithmes de traitement de signal.
- FPGA utilisés sur des cartes RF ( CAN et DAC rapides + traitement analogique associé).
- Plus de 70% (en terme de ressources) des fonctions intégrées dans le FPGA ou dans le module développé sont de nature DSP.
- Développement des " modules DSP " sur PC au travers un nombre limité d'outil ( MATLAB/simulink /QUARTUS 2).
- Capacité à modifier le projet (design) de manière rapide en intégration chez le client.

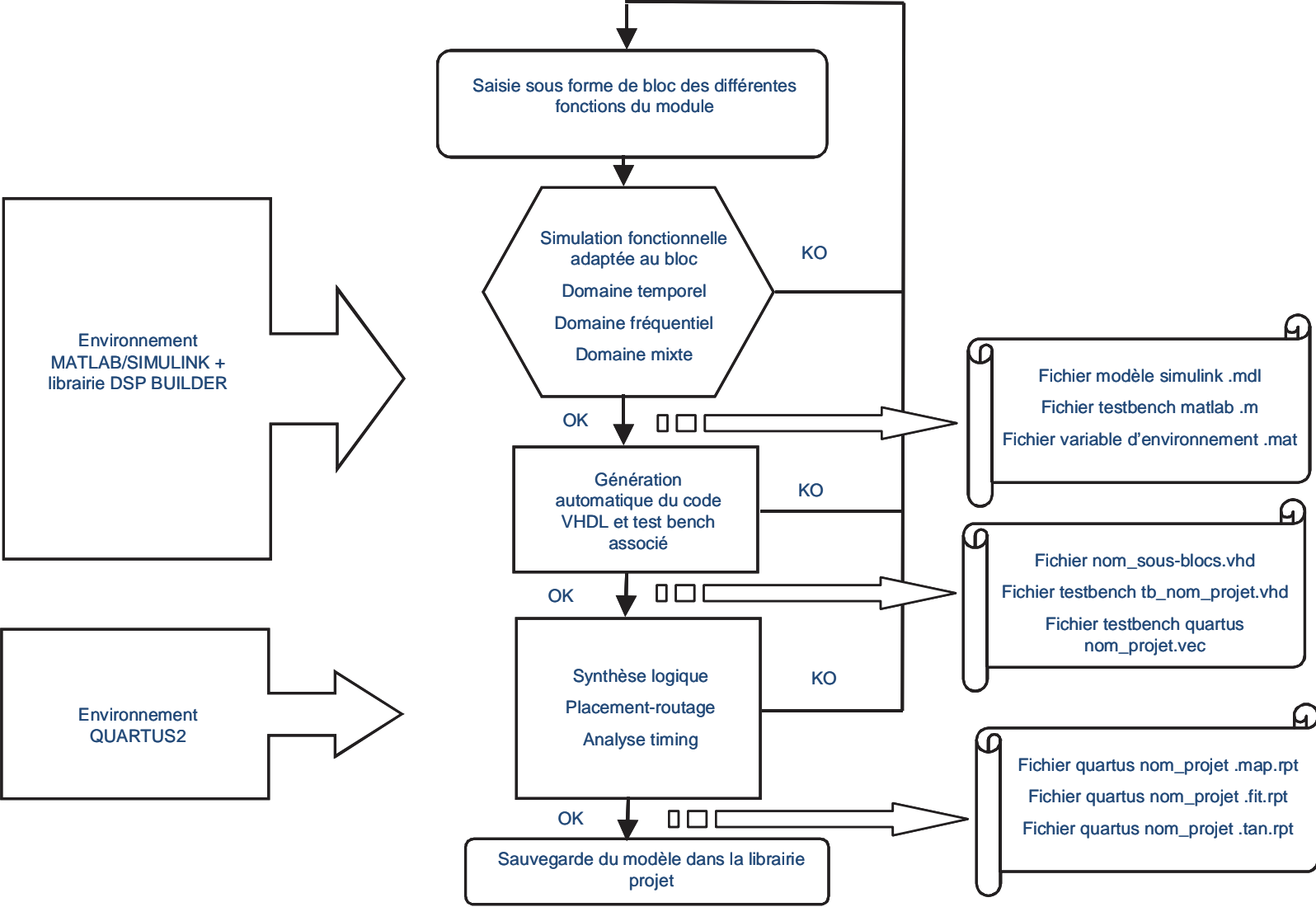


### *Nature des projets étudiés*

- Projet dont la capacité d'évolution à court terme et de façon rapide sont très importantes.
- Projet de taille moyenne permettant à un concepteur (orientés RF traitement de signal analogique et numérique) de maîtriser toute la chaîne de développement sans écriture de code VHDL.
- Projet à très forte connotation RF demandant une capacité d'analyse et de simulation des résultats dans le domaine fréquentiel et analogique.
- Mise en œuvre et caractérisation de composants mixtes rapides (CAN, CNA, PLL fractionnaire..).
- Projet faisant appel à des fonctions de traitement de signal sous forme d'IP.
- Validation d'architecture et de concept.



Processus de développement utilisé au travers l'outil DSP BUILDER



Reference - date

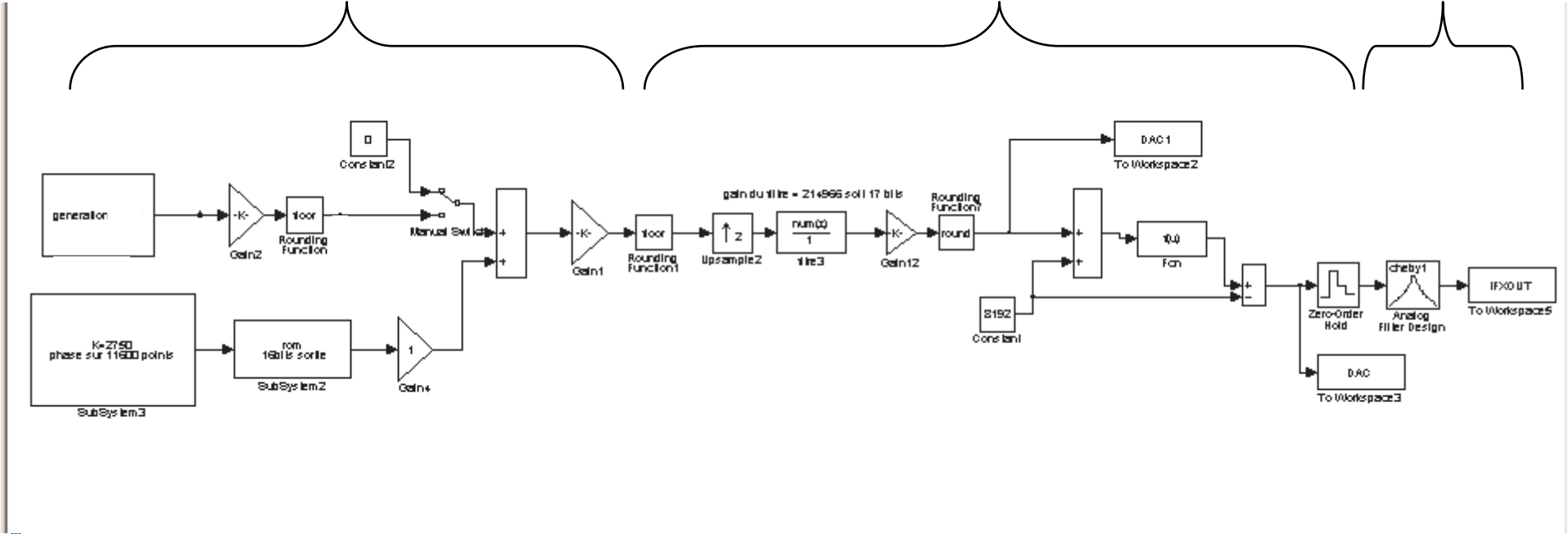


Exemple de design

Fonction DDS sous forme de bloc simulink

Fonction Interpolation et DAC sous forme de bloc simulink

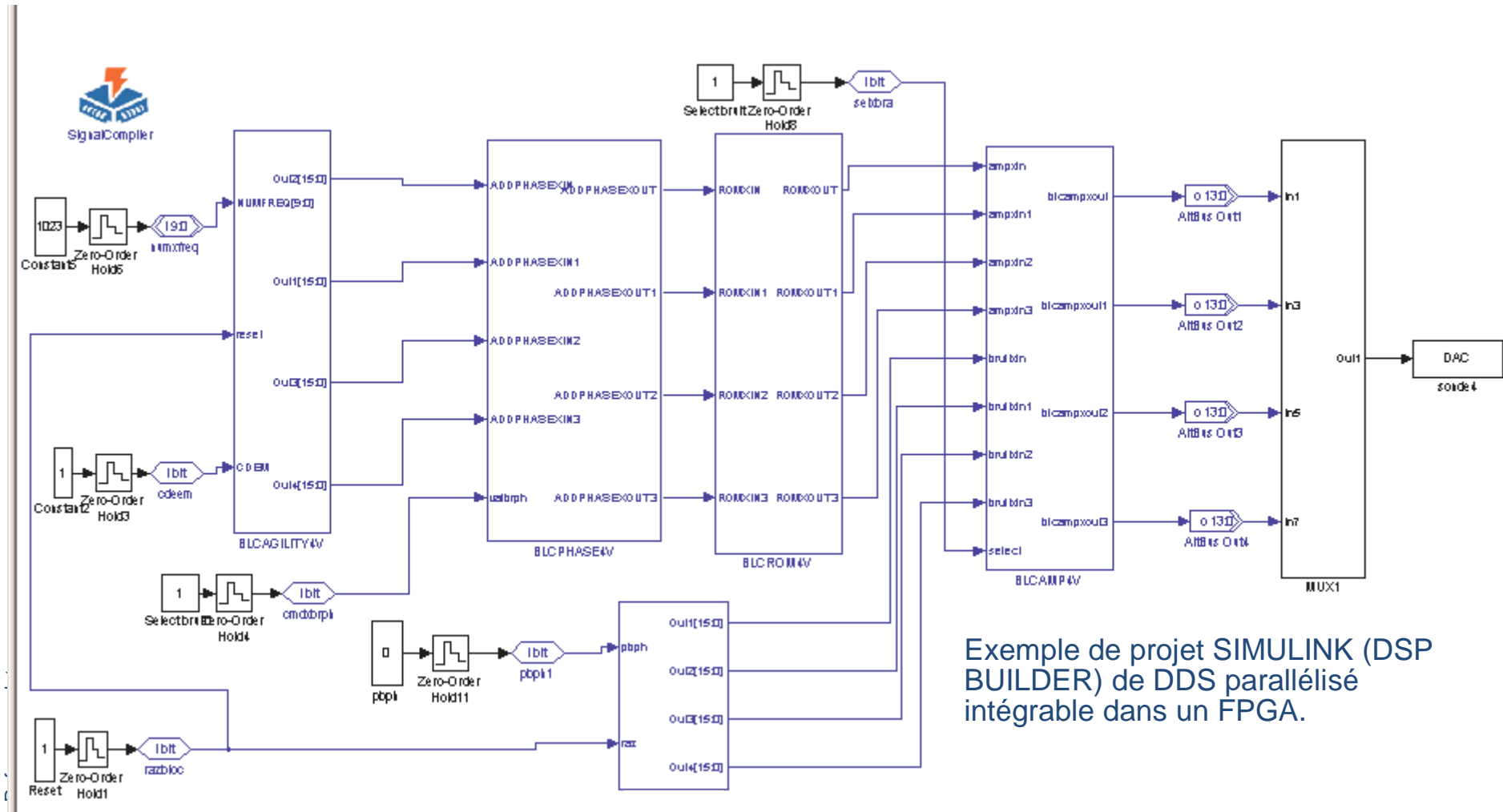
Fonction filtrage analogique sous forme de bloc simulink



Reference - da



## Exemple de design

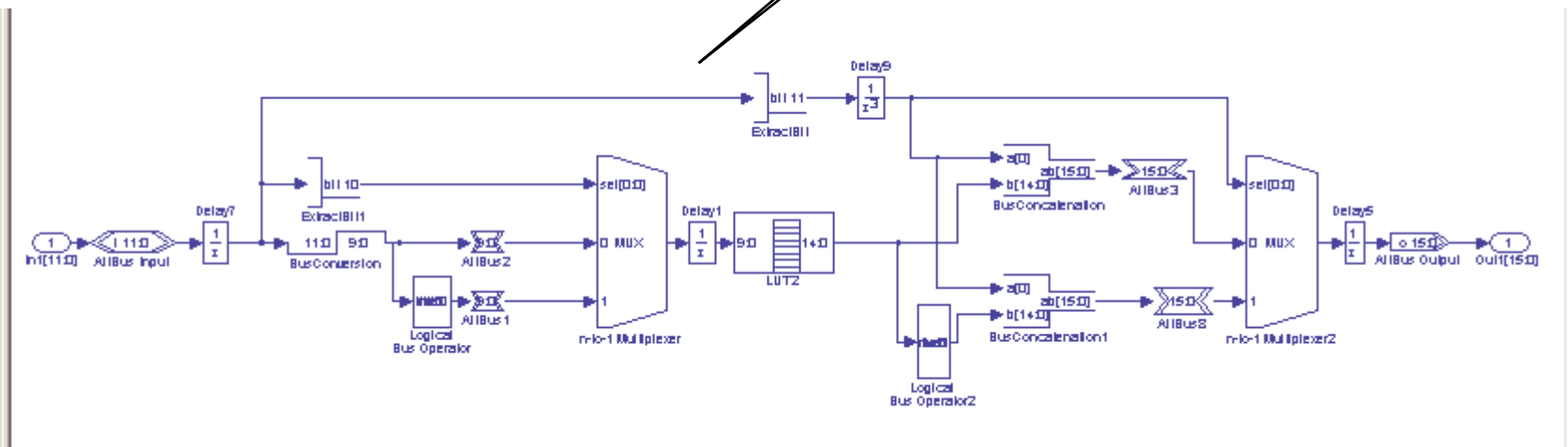
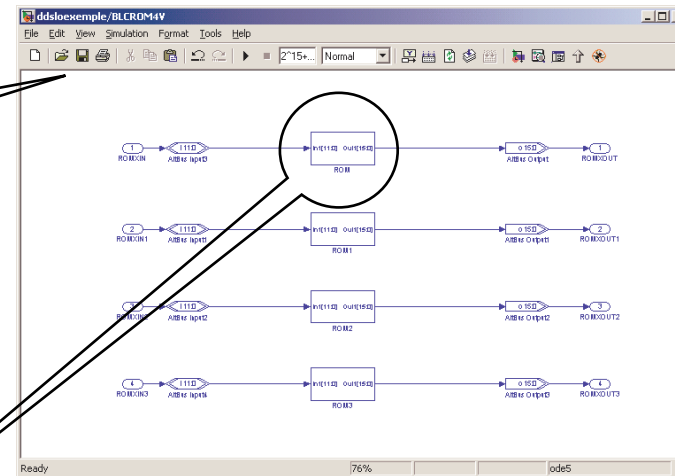
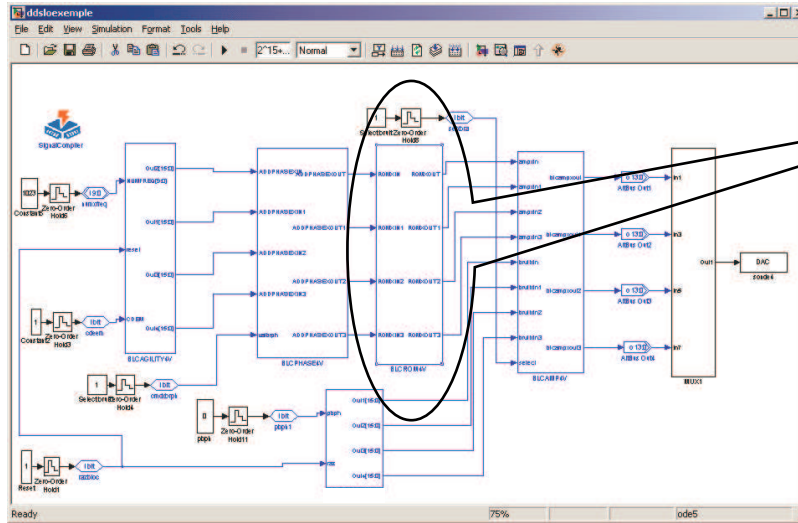


Exemple de projet SIMULINK (DSP BUILDER) de DDS parallélisé intégrable dans un FPGA.



## Exemple de design

Projet développé sous forme hiérarchique facilitant la validation du modèle



Reference - date



## *Évaluation et comparaison*

- Utilisation de l'outil DSP BUILDER depuis 5 ans sur 15 à 20 projets ciblant des FPGA de chez ALTERA (APEX, STRATIX 1 et 2, CYCLONE 1 et 2).

Outils utilisés :MATLAB R14SP3-DSP BUILDER 6.1 SP1 QUARTUS 2 6.1

- Utilisation de l'outil SYNPLIFY DSP en juin 2005 dans le cadre d'une évaluation (Mise à disposition d'une licence SYNPLIFY DSP et SYNPLIFY PRO par SYNPLICITY).

Outils utilisés :MATLAB R13SP1-SYNPLIFY DSP 2.2.0-SYNPLIFY PRO 8.1

- Utilisation en 2006 de l'outil XN GENERATOR pour concevoir un module de traitement de signal dans un VIRTEX4.

Outils utilisés :MATLAB R13SP1-XN GENERATOR 8.1-ISE 8.1

- Utilisation en 2007 de l'outil SIMULINK HDL dans le cadre d'une évaluation (Mise à disposition de l'outil par Mathworks).

Outils utilisés :MATLAB R2007 a et b



## Évaluation et comparaison

| Critères de comparaison  | Outil DSPBUILDER(6.1)<br>(ALTERA) | Outil XN GENERATOR (8.1)<br>(XILINX) | Outil SIMULINK-HDL (R2007b)<br>(MATHWORKS) | Outil Synplify DSP (2.2)<br>(SYNPLICITY) |
|--|-----------------------------------|--------------------------------------|--|--|
| Richesse de la bibliothèque et lisibilité du modèle  | ++                                | +                                    | +  | =  |
| Personnalisation et optimisation des briques de base (pipeline,format,type d'implémentation) | +                                 | +                                    | -  | =  |
| Simulation mixte (temps discret/temps continu)   | oui                               | oui                                  | partiellement                              | Non (2.2)                                |
| Indépendance du code généré et capacité à comparer des matrices fondeurs différentes         | non                               | non                                  | Oui  | Oui (Synplify pro)                       |
| Fonction automatique d'optimisation du code (folding,pipeline..)                             | non                               | non                                  | non  | oui                                      |
| Configuration minimum pour simuler (hors génération de code)                                 | Simulink/DSP BUILDER              | Simulink/XN generator/ISE            | Simulink/fixe point tool box               | Simulink/fixepoint tool box/Synplify DSP |
| Coût (hors MATLAB/SIMULINK)  | \$                                | \$                                   | \$\$\$                                     | \$\$\$                                   |

Reference - date



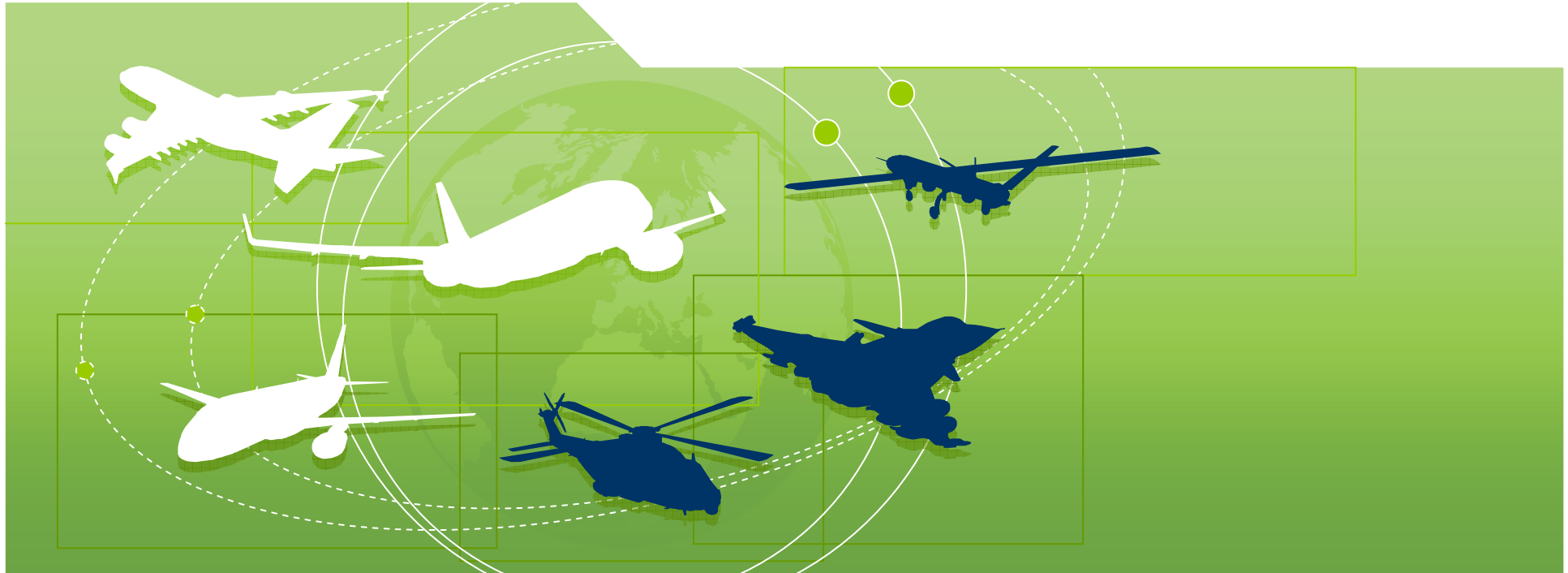
## Conclusion

### ■ Avantages

- Regroupement dans un outil haut niveau des 3 étapes spécification-conception -simulation de fonction de traitement de signal intégrable dans un FPGA
- Modélisation et simulation au sein d'un environnement de simulation mixte au niveau système.
- Simplification de la chaîne de développement et du nombre d'outils à maîtriser pour le concepteur.
- Reprise ou modification du projet par un autre concepteur rendu plus simple par simulink (conception sous forme de bloc fonctionnel).
- Réduire drastiquement l'étape lourde et sans véritable valeur ajoutée d'écriture VHDL
- Avoir la possibilité de disposer d'un code optimisé pour la matrice utilisée.
- Gagner du temps et réduire les coûts.

### ■ Inconvénients

- Disposition et bonne connaissance de l'environnement MATLAB/simulink.
- Blocs de stimulus et d'affichages peu conviviaux pour la génération et la visualisation de signaux logiques.
- Pas de flow automatique permettant de vérifier en bout de chaîne les simulations de type 'timing' et la simulation fonctionnelle issue de simulink (au travers les outils ALTERA-XILINX).



Merci de votre attention  
Questions / Réponses