Encadrants :
Laurent Fesquet,
Sylvain Huet,
Sylvain Engels,
Jean Simatic

Travaux pratiques de conception de circuit VLSI numérique

Conception d'un filtre numérique à réponse impulsionnelle finie – Simulation

19 janvier 2017

Sommaire

I) Démarrage

I.1) Connexion

Première connexion

Utilisez le mot de passe par défaut pour vous connecter au compte xph2sicxxx où xxx désigne votre numéro de binôme.

Ouvrez un terminal puis utilisez la commande `yppasswd` pour changer votre mot de passe.

Connexion ultérieures

Connectez au compte xph2sicxxx avec votre mot de passe personnalisé

I.2) Structure des répertoires

La structure des répertoires de ce TP est donnée dans le tableau ci-dessous. Vous aurez à vous reporter à ce tableau tout au long du TP pour savoir où trouver les données ou pour vous situer au bon endroit lors de l'exécution de certaines commandes.

Tableau 1 : Structure des répertoires pour le TP XPH2SICfiltre

Nom du répertoire	Description
`bench`	Contient les environnements de simulations et les programmes de tests
`config`	Contient les scripts de configuration et les fichiers d'initialisation
`doc`	Contient les documentations relatives au TP
`fpga`	Dédié à la cible FPGA
`fpga/usercontraints`	Contient le fichier des contraintes de placement et d'affectation des broches du FPGA
`fpga/par`	Dédié au placement et routage sur FPGA (Xilinx ISE)
`fpga/synth`	Dédié à la synthèse VHDL pour FPGA (Mentor Graphics Precision)
`libs`	Contient les bibliothèques VHDL compilées Lib_VHDL : bibliothèque comportementale (avant synthèse) Lib_BENCH : bibliothèque pour les modules de tests Lib_FPGA_SYNTH : synthèse pour le FPGA Lib_FPGA_PAR : placement et routage pour le FPGA Note : les répertoires correspondant apparaitront à la création de ces bibliothèques
`vhd`	Contient les sources VHDL

I.3) Avant de commencer à travailler

La première chose à faire avant de commencer à travailler est de sourcer[1] le script de configuration correspondant à l'étape du TP dans laquelle vous vous trouvez :

- `config_RTL` : configuration pour la simulation comportementale au niveau RTL
- `config_FPGA` : configuration une implantation sur FPGA

1 Dans le contexte d'une utilisation des outils de CAO dans un environnement Linux, l'expression « sourcer un script » désigne (par contamination du nom de la commande) le lancement du script à l'aide de la commande Linux `source` plutôt que par une exécution simple sur la ligne de commande.

Tous ces fichiers de configuration ont été placés dans le sous-répertoire "`config`". Ils définissent tous une variable d'environnement appelée TP_PATH qui transporte le chemin absolu de la base de l'arborescence du TP. Vous pouvez utiliser cette variable d'environnement pour exécuter vos commandes ou écrire vos scripts.

II) Prise en main du flot de conception

Ecriture d'un petit exemple

Afin de prendre en main le logiciel de simulation ModelSim, il est demandé d'écrire le code VHDL d'un compteur/décompteur 3 bits. Vous trouverez ci-dessous le code VHDL de l'entité du compteur/décompteur. Le compteur/décompteur compte de 0 à 7. Il vous est demandé d'écrire le code VHDL correspondant à l'architecture de ce circuit et de le simuler.

```
library IEEE ;
use IEEE.STD_LOGIC_1164.all ;
use IEEE.NUMERIC_STD.all ;

Entity CounterUpDn is
Port (Reset  :      in STD_LODIC ;
      Clk    :      in STD_LOGIC ;
      UpDn   :      in STD_LOGIC ;
      Q      :      out unsigned(2 downto 0));
End CounterUpDn;
Architecture A of CounterUpDn is
Begin
...
End A;
```

- Dans le répertoire `COMPTEUR`, écrire le code de l'architecture.
- Taper `source .bashrc_cpt`, puis `source config_cpt`.
- Créer une librairie pour compiler votre code avec la commande `vlib work`, puis lancer la commande `vcom counter.vhd`.
- Compléter le fichier bench.vhd et taper `vcom bench.vhd`.
- Lancer le simulateur par la commande `vsim`.
- Observer les chronogrammes.

III) Présentation et spécification du filtre numérique

Ce TP propose d'étudier et de réaliser un filtre numérique à réponse impulsionnelle fine à 32 coefficients

III.1) Étude et spécification du filtre numérique

Une étude sous Matlab a permis de spécifier la réponse fréquentielle voulue. Les courbes obtenues sont reproduites aux figures 1 et 2. Les trente-deux coefficients correspondant sont donnés au tableau 2.

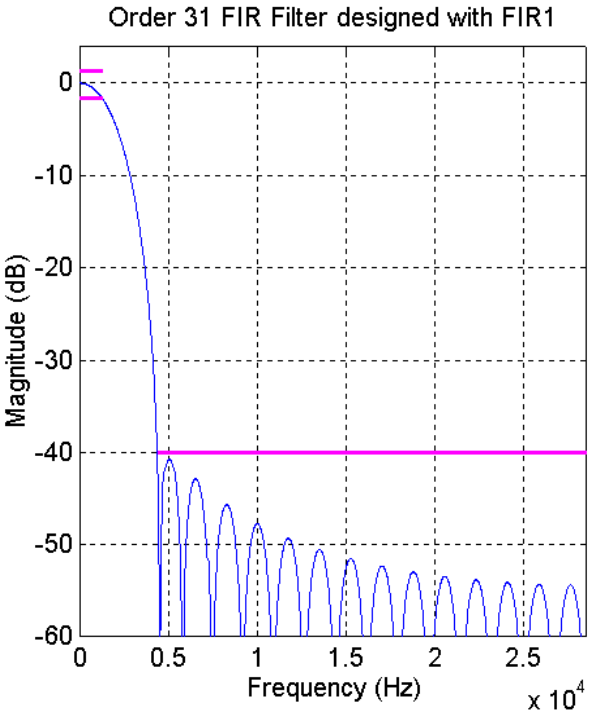


Fig 1 : Réponse fréquentielle du filtre numérique à 32 coefficients

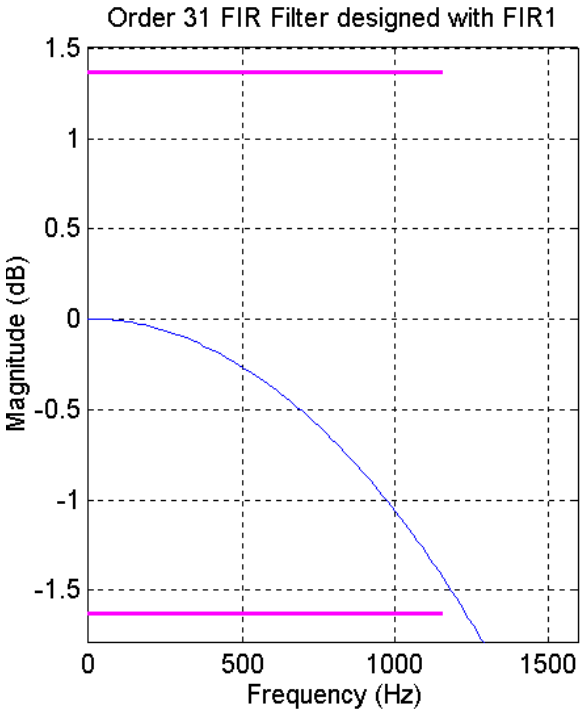


Fig 2 : Réponse fréquentielle du filtre numérique à 32 coefficients (zoom)

Tableau 2 : Liste des 32 coefficients du filtre numérique

#	Binaire	Hexa-décimale	Décimale arrondie	Décimale	Erreur
0	**0 0 0 0 1 1 0 1**	**0D**	0.0508	0.0512	0.0004
1	**0 0 0 1 0 1 0 1**	**15**	0.0820	0.0832	0.0012
2	**0 0 0 1 1 1 1 1**	**1F**	0.1211	0.1245	0.0034
3	**0 0 1 0 1 1 0 0**	**2C**	0.1719	0.1754	0.0035
4	**0 0 1 1 1 1 0 0**	**3C**	0.2344	0.2355	0.0011
5	**0 1 0 0 1 1 0 1**	**4D**	0.3008	0.3039	0.0031
6	**0 1 1 0 0 0 0 1**	**61**	0.3789	0.3791	0.0002
7	**0 1 1 1 0 1 0 1**	**75**	0.4570	0.4591	0.0020
8	**1 0 0 0 1 0 1 0**	**8A**	0.5391	0.5412	0.0021
9	**1 0 0 1 1 1 1 1**	**9F**	0.6211	0.6226	0.0015
10	**1 0 1 1 0 0 1 1**	**B3**	0.6992	0.7001	0.0009
11	**1 1 0 0 0 1 0 1**	**C5**	0.7695	0.7707	0.0012
12	**1 1 0 1 0 1 0 0**	**D4**	0.8281	0.8314	0.0033
13	**1 1 1 0 0 0 0 1**	**E1**	0.8789	0.8795	0.0006
14	**1 1 1 0 1 0 0 1**	**E9**	0.9102	0.9128	0.0026
15	**1 1 1 0 1 1 1 0**	**EE**	0.9297	0.9298	0.0001
16	**1 1 1 0 1 1 1 0**	**EE**	0.9297	0.9298	0.0001
17	**1 1 1 0 1 0 0 1**	**E9**	0.9102	0.9128	0.0026
18	**1 1 1 0 0 0 0 1**	**E1**	0.8789	0.8795	0.0006
19	**1 1 0 1 0 1 0 0**	**D4**	0.8281	0.8314	0.0033
20	**1 1 0 0 0 1 0 1**	**C5**	0.7695	0.7707	0.0012
21	**1 0 1 1 0 0 1 1**	**B3**	0.6992	0.7001	0.0009
22	**1 0 0 1 1 1 1 1**	**9F**	0.6211	0.6226	0.0015
23	**1 0 0 0 1 0 1 0**	**8A**	0.5391	0.5412	0.0021
24	**0 1 1 1 0 1 0 1**	**75**	0.4570	0.4591	0.0020
25	**0 1 1 0 0 0 0 1**	**61**	0.3789	0.3791	0.0002
26	**0 1 0 0 1 1 0 1**	**4D**	0.3008	0.3039	0.0031
27	**0 0 1 1 1 1 0 0**	**3C**	0.2344	0.2355	0.0011
28	**0 0 1 0 1 1 0 0**	**2C**	0.1719	0.1754	0.0035
29	**0 0 0 1 1 1 1 1**	**1F**	0.1211	0.1245	0.0034
30	**0 0 0 1 0 1 0 1**	**15**	0.0820	0.0832	0.0012
31	**0 0 0 0 1 1 0 1**	**0D**	0.0508	0.0512	0.0004

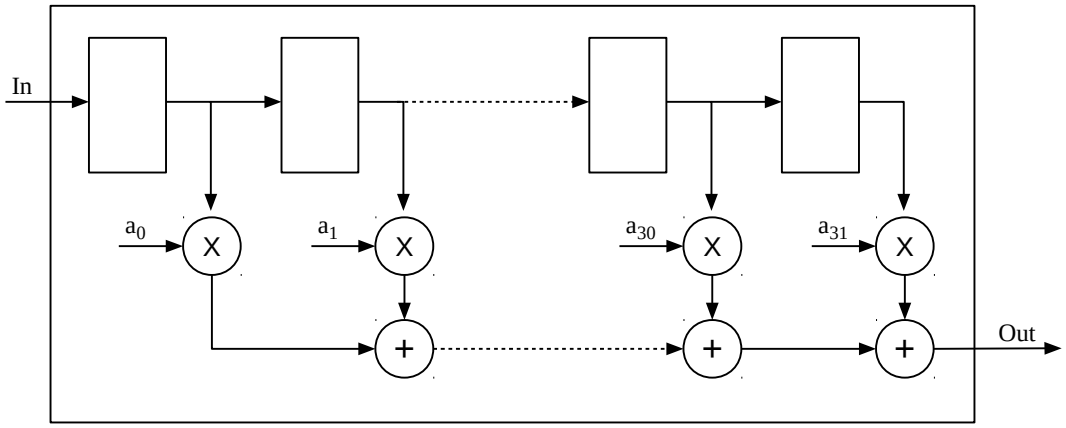


Fig 3 : Architecture canonique d'un filtre numérique RIF

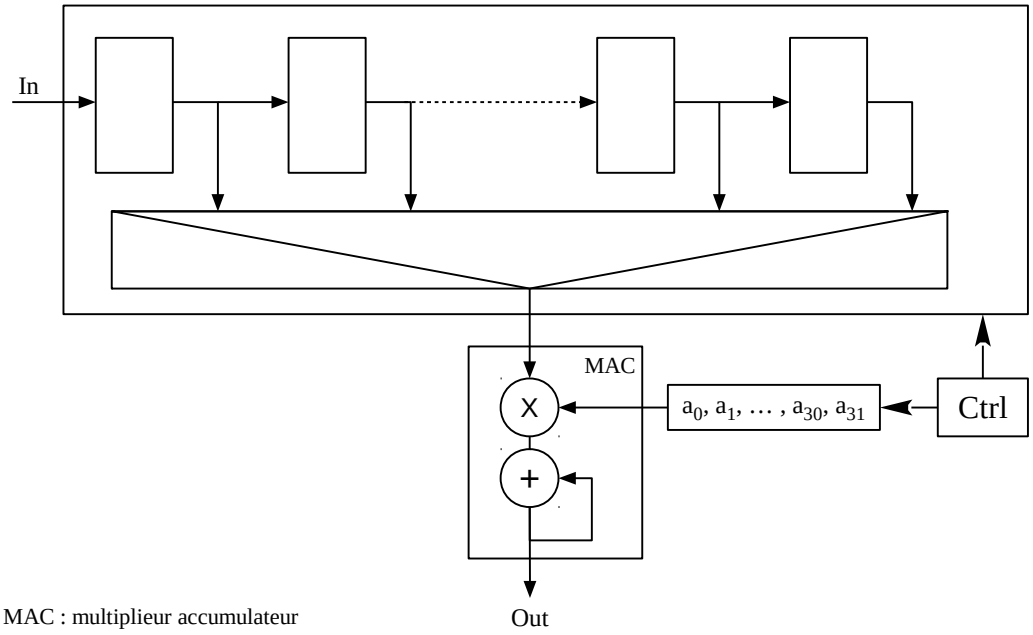


Fig 4 : Architecture optimisée en taille pour un filtre numérique RIF

Cette architecture est détaillée à la figure 5. Sa description en VHDL est consignée par l'ensemble des fichiers du répertoire `vhd`.

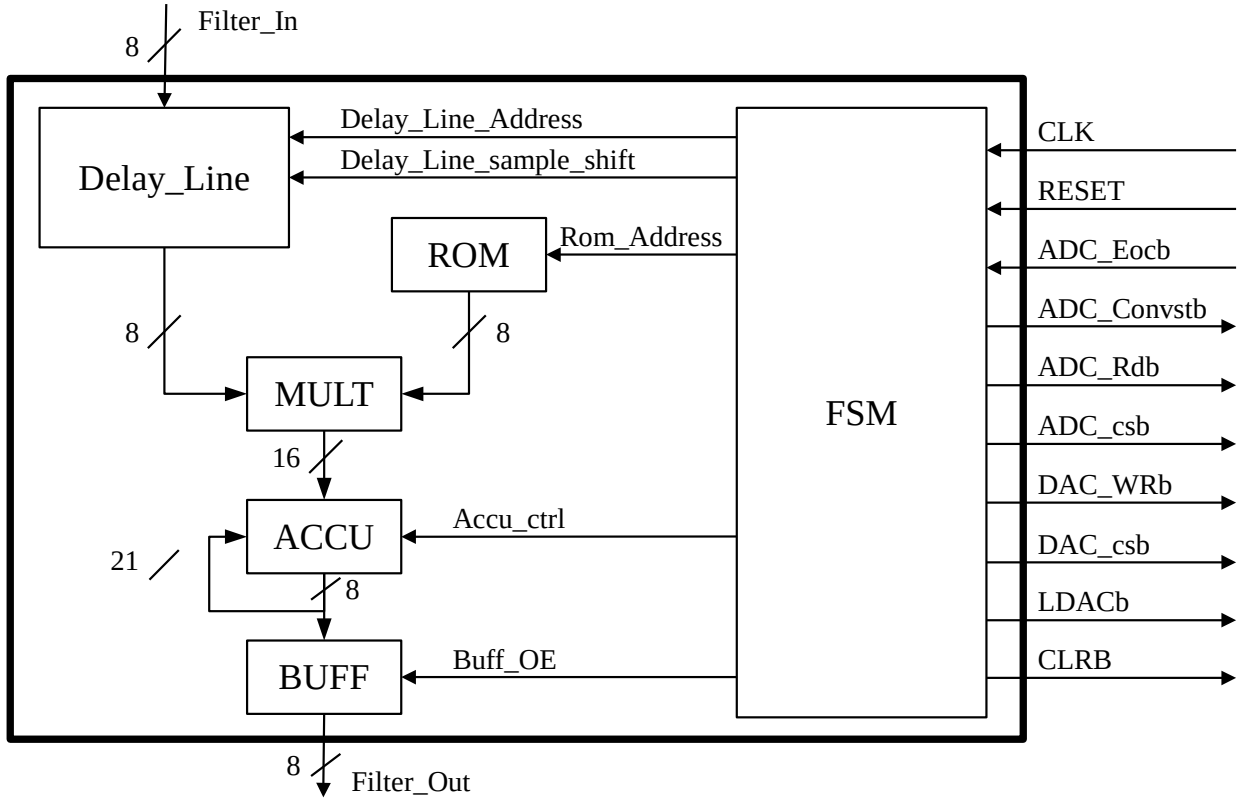


Fig 5 : Architecture détaillée de filtre RIF

III.2) Travail demandé

a) Comprendre et commenter l'ensemble du code source donné dans les répertoires "`vhd`" et "`bench`".

b) Faire un chronogramme spécifiant le fonctionnement du système.

c) Extraire la machine à états du chronogramme. La dessiner sous forme de graphe de transitions.

Compléter le fichier "`vhd/fsm.vhd`". Il s'agit d'écrire l'architecture associée à l'entité `fsm` du contrôleur du filtre. Le contrôleur doit être tel que le filtre puisse s'interfacer avec le système de conversion analogique/numérique et numérique/analogique (cf. annexe).

d) Modifier et compléter le programme de test.

Le fichier de test "`bench/bench.vhd`" simule le fonctionnement des convertisseurs. Il fournit des signaux en entrée du filtre et consomme ceux en sortie en tenant compte des caractéristiques fonctionnelles et temporelles des signaux allant ou provenant des convertisseurs (cf. documentation technique en annexe).

Ce programme de test, que vous avez à écrire, devra vérifier que le filtre est fonctionnellement correct en appliquant les stimulus appropriés de façon à examiner la réponse impulsionnelle, la réponse indicielle ou la réponse à une somme de sinusoïdes. Dans un premier temps, il faudra modéliser les convertisseurs et vérifier leur bon comportement temporel à l'aide de clauses `assert`. Puis, il faudra fournir les stimulus appropriés pour tester (toujours à l'aide de clauses `assert`) les différentes réponses. Expliquez votre démarche.

e) Simuler le filtre complet dans l'environnement de test développé pour vérifier son bon fonctionnement. Argumenter avec précision pourquoi le fonctionnement du filtre est correct.

Pour effectuer des simulations comportementales avec le logiciel ModelSim, vous devez au préalable sourcer le script de configuration `config_RTL`. Le mini guide fourni sur ModelSim donne les principales commandes à connaitre pour utiliser ce logiciel. N'hésitez pas à consulter aussi l'aide du logiciel (le mini guide indique où trouver cette aide).

Un compte rendu synthétique, décrivant l'ensemble de vos travaux, est à rendre à la fin de la deuxième séance.

IV) Réalisation du filtre sur FPGA

Pour cette partie vous devez sourcer le fichier de configuration `config_FPGA`.

IV.1) Synthèse de la description VHDL avec l'outil Precision de Mentor Graphics

IV.1.1) Lancer l'outil Précision

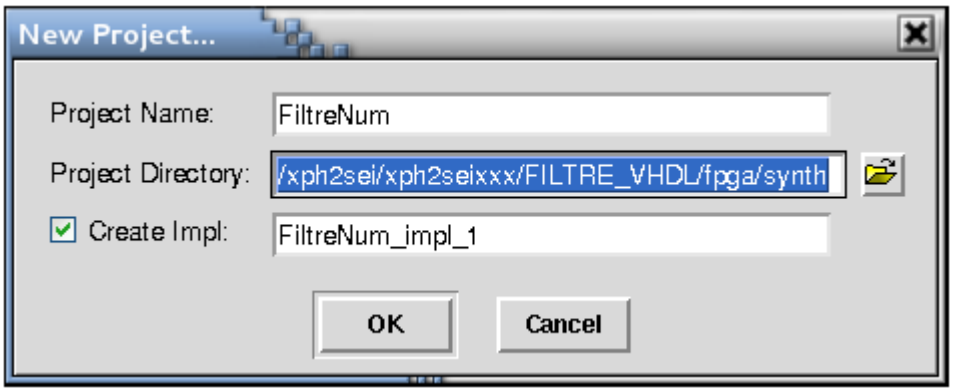
a) Sourcer le script de configuration `config_FPGA` et assurer vous que vous êtes dans le répertoire `${TP_PATH}/fpga/synth`

b) Démarrer l'outil en exécutant la commande
`precision &`

c) Créer un nouveau projet via le menu « Project | New Project »

- Donner le nom de votre choix à votre projet, par exemple « FiltreNum », en renseignant la zone de saisie « Project Name ».
- Assurer vous que le champ « Project Directory » désigne bien le répertoire `${TP_PATH}/fpga/synth`.
- La zone de saisie « Create Impl » donne le nom du répertoire qui contiendra la description du circuit synthétisé. Le nom par défaut est le nom du projet (champ « Project Name ») suivie de « impl_1 » (adapter ce nom si besoin est). La case à cocher doit être cochée pour que le répertoire soit créé.

a) Cliquer OK.
Le sous répertoire destiné à contenir la description du circuit synthétisé est alors créé.



IV.1.2) Choix de la technologie cible :

A l'aide de l'icône « Setup Design », ouvrir la fenêtre « Project Settings ». Il peut être nécessaire de cliquer l'onglet « Design » dans la colonne de gauche pour voir cette icône.

Choisir :

- Technology : Xilinx Spartan3
- Device : 3s200ft256
- Speed Grade : -5

Cocher `Set Frequency` et donner la fréquence de cadencement de la carte cible (50 MHz). Laisser les autres options par défaut.

Valider (OK) et sauvegarder le projet.

IV.1.3) Définir les modèles à synthétiser

L'icône « Add Input Files » (colonne de gauche onglet « Design »), permet de spécifier les fichiers qui contiennent les entités et les architectures à synthétiser.

Se placer dans `${TP_PATH}/vhd` et choisir tous les fichiers source du filtre (fichiers *.vhd).

Dans Precision, le fichier contenant le module englobant (*top module*) du projet est mis en gras et en bas de la liste « Input Files » dans la zone « Project Files ». Dans ce projet, le module englobant est contenu dans le fichier `filter.vhd`. Il faut donc le désigner comme tel. Pour cela, sélectionner le fichier `filter.vhd` cliquer droit et sélectionner « Move File to Bottom ».

IV.1.4) Options

Il est possible d'optimiser la synthèse, soit en minimisant la surface, soit en améliorant les performances temporelles. Pour accéder aux options d'optimisation sélectionner le menu « Tools | Options | Optimization Settings ».

IV.1.5) Spécifier le format de sortie du résultat de synthèse :

Le menu « Tools | Options | Output Options » permet de définir le format de sortie de la liste de porte (*netlist)* obtenue après synthèse ainsi que le nom du fichier dans lequel la synthèse sera enregistrée.

Spécifier le nom du fichier de sauvegarde dans le champ « Output File Base Name » : `filtre_synth.`
Sélectionner les formats EDIF et VHDL.
Décocher l'option « Generate Vendor Constraint File ».

Avec ces choix la synthèse produira deux fichiers :

- `filtre_synth.edf` nécessaire à l'étape de placement et routage qui suit
- `filtre_synth.vhd` destiné à la simulation après synthèse

Ces fichiers seront enregistrés dans le sous répertoire `FiltreNum_impl_1` selon notre exemple.

IV.1.6) Exécution de la synthèse

La synthèse peut être lancée à partir des icônes « Compile », pour la première étape, et « Synthesize » pour la synthèse finale. Il peut être nécessaire de cliquer l'onglet « Design » dans la colonne de gauche pour voir ces icônes.
Le compte-rendu de ces opérations apparait dans l'onglet « Transcript ».

IV.1.7) Visualisation du résultat de la synthèse :

Les résultats de la synthèse sont visualisable par le biais des icônes de l'onglet « Schematics » dans la colonne de gauche :

- L'icône « View RTL Schematic » permet de visualiser la vue RTL de la modélisation VHDL.
- L'icône « View Tech Schematic » permet de visualiser le schéma de portes obtenue avec la bibliothèque de portes logiques de la technologie ciblée.
- L'icône « View Critical Path » permet de visualiser le chemin critique dans le circuit synthétisé.

Pour connaitre les statistiques de performances du circuit synthétisé, cliquer sur l'icône « Area report » pour le taux d'occupation du FPGA et sur l'icône « Report Timing » pour les timings. Ces icônes deviennent visibles en cliquant sur l'onglet « Design Analysis » dans la colonne de gauche

IV.1.8) Enregistrement de la synthèse

Le résultat de la synthèse est conservé dans le fichier `filtre_synth.edf`.
Vérifiez que les fichiers `filtre_synth.edf` et `filtre_synth.vhd` sont bien présents dans le répertoire créé pour la synthèse (`FiltreNum_impl_1` dans notre exemple).
Procéder ensuite à l'enregistrement du projet en sélectionnant « Project | Save Project » avant de quitter l'outil Precision.

IV.1.9) Simulation après synthèse

En vous inspirant du script de compilation compil_VHDL, créez un nouveau script pour une compilation avec Modelsim du fichier VHDL du circuit synthétisé. Pensez à mettre à jour vos bibliothèques.

IV.2) Placement et routage ciblés FPGA avec l'outil ISE de Xilinx

IV.2.1) Le fichier des contraintes de l'utilisateur

Même si un FPGA est très flexible, il n'est pas possible d'affecter n'importe quel signal à n'importe quelle broche. En effet, les broches sont regroupées en ports qui peuvent être associés à des fonctions spéciales comme un port sériel, un port JTAG ou un décodeur pour un afficheur. Même dans le cas le plus courant d'entrées/sorties tout ou rien à usage général, deux ports différents n'acceptent pas forcément la même plage de tension (par exemple un domaine en 0-5V et un autre en 0-3V), la sortance peut être différente et certaines terminaisons sont soit en entrée soit en sortie alors que d'autres sont bidirectionnelles. En outre, certaines broches sont dédiés aux alimentations et au moins une est réservée pour recevoir un signal d'horloge. Lorsque le FPGA est déjà câblé sur une carte électronique prête à l'emploi, comme celle utilisée dans ce TP, les contraintes sur l'affectation des ports et des broches sont encore plus importantes. Par exemple, une entrée tout ou rien initialement bidirectionnelle sur le FPGA nu, peut, une fois le FPGA câblé, être désormais en entrée uniquement (cas d'un bouton poussoir) ou en sortie seulement (cas d'une LED).

Il faut donc renseigner l'outil de placement et routage sur la correspondance qu'il doit respecter entre les ports physiques du FPGA et les ports formels définis par l'entité du module englobant dans la description matérielle. Il faudra également spécifier la direction pour les ports bidirectionnels, la plage de tension, la sortance, etc. Toutes ces informations sont consignées dans le fichier des contraintes de l'utilisateur (*User Constraint File* – UCF).

Dans le cas de ce TP, le fichier des contraintes de l'utilisateur a été préparé pour vous et placé dans le répertoire `${TP_PATH}/fpga/userconstraints`.

IV.2.2) Lancement d'ISE et paramétrage du projet

Rappel : vous devez avoir sourcé le script de configuration `config_FPGA` avant de pouvoir utiliser l'outil.

- Se placer dans le répertoire `${TP_PATH}/fpga/par` et démarrer l'outil en exécutant la commande `ise &`
- Sélectionner le menu « Project | New Project »
- Dans la fenêtre « New Project Wizard – Create New Project », renseigner
  - le nom du projet pour le placement et le routage, par exemple « FiltreNum »
  - l'emplacement du répertoire pour le placement et le routage.
    Indiquer le chemin `${TP_PATH}/fpga/par`
  - sélectionner « EDIF » dans la liste déroulante « Top-level source type »
  - cliquer « Next »
- Dans la fenêtre « New Project Wizard – Import EDIF/NGC Project », indiquer
  - le fichier edf créer lors de la synthèse (cf. § 8), dans la zone de saisie « Input Design »
  - le fichier `${TP_PATH}/fpga/userconstraints/filtre.ucf` dans la zone de saisie « Constraint file »
  - Décocher la case « Copy the constraints file to the project directory » de façon à ce que le fichier des contraintes de l'utilisateur *.ucf ne soit pas dupliqué.
  - cliquer « Next »
- Dans la fenêtre « New Project Wizard – Device Properties », les caractéristiques du FPGA (Family, *Device, Package, Speed*) doivent être automatiquement remplies.
  S'assurer que le champ « Preferred language » est positionné sur VHDL.
- Cliquer « Next » puis « Finish »

IV.2.3) Placement et routage

Les commandes de placement et routage sont regroupées dans le panneau de gauche « Processes » au sein d'une arborescence appelée « Implement Design » (voir Fig 6).

Les commandes de placement et routage donnent lieu à des rapports qui sont accessibles en dépliant l'arborescence.

Le rapport général (*Summary Report*) qui peut être visualisé par l'onglet « Design Summary » dans le panneau de droite, existe aussi sous forme d'un fichier HTML (`FILTER_summary.html`) dans le répertoire du projet de placement routage. Il est ainsi possible de consulter ce rapport sans devoir relancer l'outil.

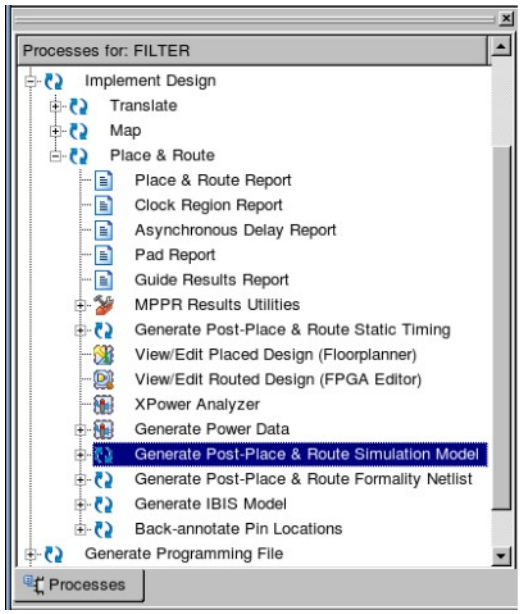


Fig 6 : Génération des modèles de simulation temporelle

Génération des modèles de simulation temporelle

Double-cliquer sur la ligne « Generate Post-place & Route Simulation Model » dans le panneau « Processes ».

Emplacement des fichiers produits pour la simulation temporelle

En supposant que l'entité du module englobant (*top module*) ait été appelée « FILTER » et que le nom du projet pour le placement et le routage soit « FiltreNum », les fichiers de timing nécessaires à la simulation après placement et routage sont automatiquement baptisés `FILTER_timesim.sdf` et `FILTER_timesim.vhd`. Ils sont générés dans le sous répertoire :

`${TP_PATH}/fpga/par/FiltreNum/netgen/par/`

IV.2.4) Génération du fichier de configuration du FPGA

Cette opération va créer un fichier qui sera chargé dans le FPGA pour changer sa configuration de façon à refléter le circuit que vous venez de placer et router. Ce type de fichier est appelé *bitstream* en anglais.

Double-cliquer sur la ligne « Generate Programming File » dans le panneau « Processes ».

IV.2.5) Simulation après placement et routage

Créez un nouveau script pour une compilation avec Modelsim pour le circuit routé. Le fichier VHDL à compiler est celui créer à l'étape 10.

Le paquetage VITAL pour le FPGA utilisé dans ce projet a été préalablement précompilé. Il est introduit dans le projet via la bibliothèque logique `simprim` qui est définie dans le fichier `modelsim.ini`.

Pour que la simulation après placement et routage prenne tout son sens, elle doit être rétroannotée avec les informations temporelles contenues dans le fichier sdf généré lors du placement et routage (cf. § 10). Pour cela il faut charger le fichier de rétroannotation dans le simulateur Modelsim. Ceci peut être à partir de l'interface graphique ou bien au démarrage sur la ligne de commande.

i) Chargement du fichier de rétroannotation par l'interface graphique de Modelsim

- Dans ModelSim, sélectionnez le menu « Simulate | Start Simulation… ».
- Dans l'onglet « Design » de la fenêtre « Start Simulation » qui vient d'apparaitre, sélectionnez votre bibliothèque de test, votre entité de test et l'architecture de test que vous voulez utiliser. Vous noterez que la zone « Design Unit » se remplit automatiquement avec les éléments sélectionnés dans la liste. Vous pouvez directement écrire dans cette zone à condition de respecter la syntaxe

  `Bibliothèque.Entité(architecture)`
- La liste « Resolution » sert à sélectionner une résolution temporelle raisonnable avec la période des signaux à analyser. Inutile d'exiger la picoseconde avec une période d'horloge de 20 ns !
- Dans la zone « Optimization », cocher « Enable Optimization » et cliquer le bouton « Optimization Options… »

- Dans l'onglet « Visibility » de la fenêtre « Optimization Options », sélectionner « Add full visibility to all modules » et cliquer OK.
- Dans l'onglet « SDF » de la fenêtre « Start Simulation », zone « SDF Files », cliquez le bouton « Add »
- Dans la fenêtre « Add SDF Entry » qui est apparue, renseignez le nom complet du fichier SDF à ajouter. C'est le fichier généré lors du placement et routage (cf. § 10)
- De retour dans l'onglet «SDF» de la fenêtre « Start Simulation », cochez les cases « Disable SDF Warnings » et « Reduce SDF Errors to Warnings » de la zone « SDF Options »

ii) Chargement du fichier de rétroannotation par la ligne de commande

La ligne de commande à saisir doit être de la forme
`vsim [options] Bibliothèque.Entité(architecture)`
Les options à utiliser ici sont les suivantes :

`-sdftyp` : pour indiquer quelle instance de quelle entité doit être simulée avec quel fichier SDF, suivant le format
/Entité/Instance={fichier SDF}
ex : pour simuler une instance `U1` d'une entité `MonBench` avec le fichier `FILTER_timesim.sdf` du répertoire `${TP_PATH}/{répertoire des SDF}/`, la commande sera
`vsim -sdftyp /MonBench/U1=${TP_PATH}/{répertoire des SDF}/FILTER_timesim.sdf`

`-sdfnoerror` : pour désactiver les erreurs SDF

`-sdfnowarn` : pour désactiver les mises en garde SDF

`-voptargs=+acc` : pour donner une visibilité complète à tous les signaux de tous les modules

iii) Simulation

La simulation s'effectue de la même façon que pour les simulations avant et après synthèse.

Dès que la simulation après placement et routage donne satisfaction, vous pouvez passer à l'implantation sur la carte FPGA.

IV.3) Programmation de la carte FPGA Xilinx Spartan3

Se reporter aux annexes pour connaitre l'emplacement et le branchement des connecteurs sur le banc de test.

IV.3.1) Configuration du FPGA

L'outil de programmation des FPGA Xilinx s'appelle iMPACT.

Charger la configuration de l'outil en sourçant les scripts suivants :

```
/softslin/configCAO/Xilinx/config_ISE
/softslin/configCAO/Xilinx/settings_xilinx_LD_PRELOAD.sh
```

Pour transférer un fichier de configuration dans un FPGA Xilinx, procéder comme suit :

- Mettre sous tension la carte *Spartan 3 Starter Board.*
- Dans un terminal, taper la commande `impact`.
- Valider par « OK », le message d'information disant que le répertoire de travail a été changé et a été positionné sur le répertoire courant.
- Dans la fenêtre « iMPACT Project », sélectionner « create a new project » et désigner le répertoire où créer le projet (`${TP_PATH}/fpga/par`) grâce au bouton « Browse ». Cliquer « OK ».

- Dans la fenêtre « iMPACT – Welcome to iMPACT », s'assurer que « Configure devices using Boundary-Scan (JTAG) » est sélectionné et que la liste déroulante est positionnée sur « Automatically connect to a cable and identify Boundary -Scan chain » puis cliquer « Finish ».
  Si le message « Warning:iMPACT:923 – cable not found » apparait, c'est que le port JTAG n'est pas connecté au port USB du PC (attention au sens du câble) ou que la carte FPGA n'est pas sous tension ou que le pilote USB n'est pas chargé (cf. ci-dessus les scripts à sourcer).
- Une fenêtre « Assign New Configuration File » apparait pour indiquer la configuration du composant xc3s200 : c'est le FPGA. Parcourir l'arborescence pour retrouver le fichier de configuration généré au §IV.2.4 (*bitstream* d'extension *.bit). Sélectionner ce fichier puis cliquer « Open ».
- La fenêtre « Assign New Configuration File » réapparait pour la configuration d'un composant xcf02s : c'est une EEPROM présente sur la carte FPGA mais qui n'est pas utilisée dans ce TP. Continuer en cliquant « Bypass » puis valider par OK le message d'information suivant.
- Dans l'onglet « Boundary Scan » de la fenêtre d'iMPACT (cf. Fig 7), faire un clic droit sur l'icône du FPGA xc3s200, puis sélectionner « Program »
- Le FPGA est reconfiguré en une dizaine de secondes.

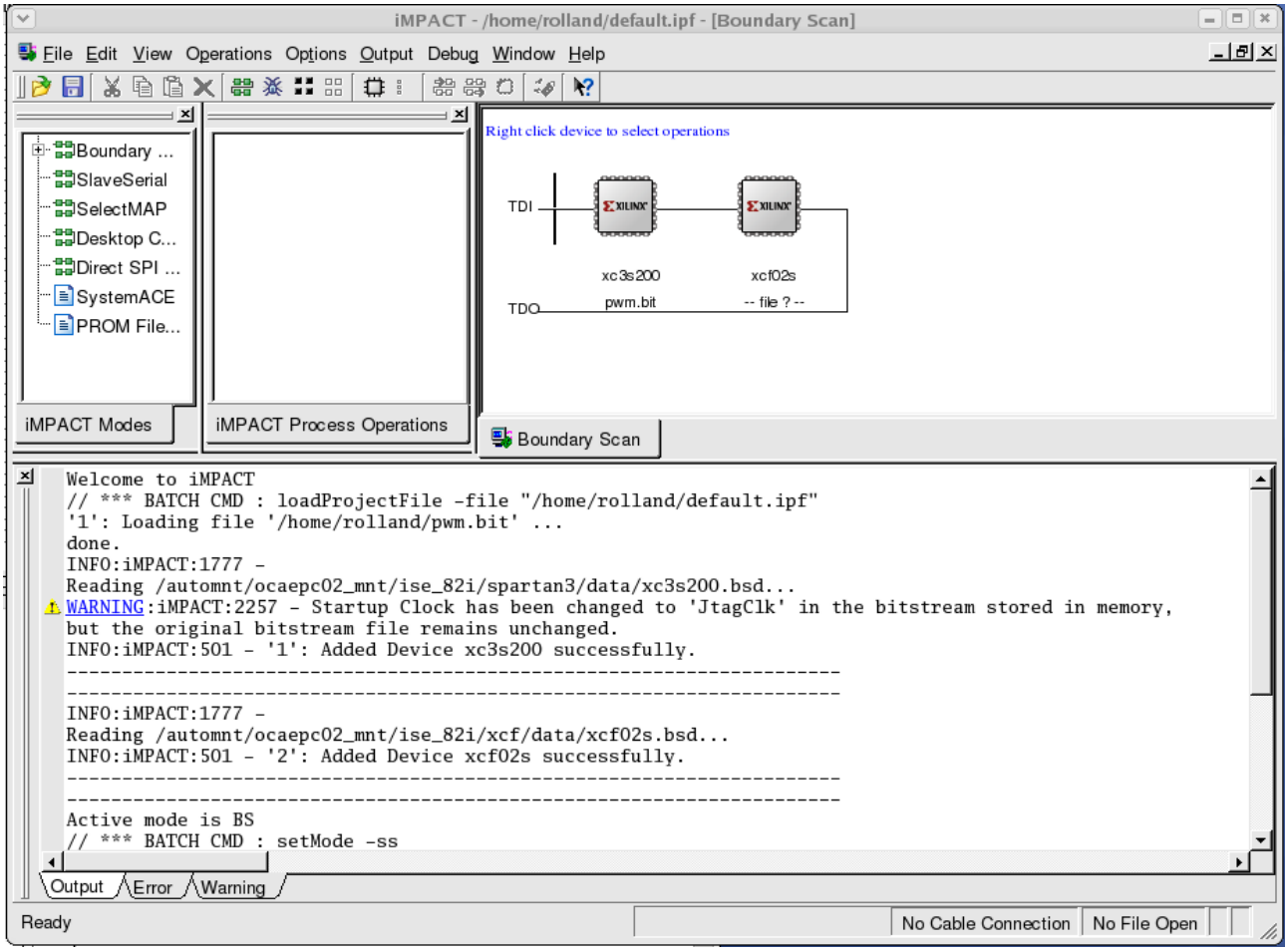


Fig 7: La fenêtre de l'outil iMPACT

IV.3.2) Test du circuit

En vous aidant du schéma d'implantation simplifié et de la photo du banc de test matériel (cf. § IV des annexes), assurez vous que le générateur basse fréquence (Géné BF) est bien connecté à l'entrée `AnaIn` de la chaine A et renvoyée sur une des entrées de l'oscilloscope. Vérifiez que la sortie `AnaOut` de la chaine A est reliée à l'oscilloscope. Visualisez les signaux de contrôles du CAN et du CNA de la chaine A.

À l'aide du Géné BF et de l'oscilloscope, contrôlez le fonctionnement du filtre numérique.

Afin de lever les doutes sur la chaine de numérisation, en cas de non fonctionnement de votre circuit, les fichiers de configuration de référence suivants sont mis à votre disposition dans le répertoire `/softslin/configCAO/Xilinx` :

- le fichier `test_sans_filtre_spartan3.bit` valide le fonctionnement de la chaine de numérisation sans filtrage
- le fichiers `valid_cna_spartan3.bit` valide le fonctionnement du convertisseur numérique vers analogique

ATTENTION : s'il n'est pas possible de faire fonctionner le banc de test avec les fichiers de configuration de référence indiqués ci-dessus, réinitialiser la carte FPGA en débranchant le connecteur d'alimentation, le port JTAG et le Géné BF.

V) Travail demandé

V.1) Travaux à réaliser et questions à traiter

a) Réaliser la synthèse du filtre à partir de l'ensemble du code source du répertoire `vhd` sur la technologie Xilinx Spartan3.

b) Commenter les différences entre la technologie FPGA et la technologie ASIC. Est-ce que le résultat est conforme à votre intuition ? commenter.

c) Résultat est conforme à votre intuition ? commenter.

d) A quoi correspond l'opération « compile » dans Precision ?

e) Qu'effectue l'outil Precision lors de la phase « synthesize » ? Lors de la phase « optimize » ?

f) Réaliser le placement et routage sur le FPGA Spartan3 de Xilinx.

g) Regarder le routage obtenu dans Xilinx FPGA Editor.

h) Que dire du nombre de LUTs utilisées après placement-routage ? Qu'en est-il du nombre de registres ?

i) À quoi sert le fichier `ucf` ? Que contient-il ?

j) Simuler le filtre complet obtenu après placement et routage, toujours dans l'environnement de test développé, mais en tenant compte des délais de propagation introduits dans les portes et les interconnexions. Vérifier que le fonctionnement du filtre reste correct.

k) Qu'est-ce que le paquetage VITAL ? À quoi sert-il ?

l) Effectuer la validation du filtre sur la carte de test en chargeant le fichier de configuration (fichier `*.bit`) dans le circuit Xilinx à programmer.

m) Produire un compte-rendu final.

V.2) Compte-rendu final

Un compte-rendu décrivant l'ensemble de vos travaux sera à rendre à la fin de la quatrième séance. Il devra en particulier inclure les indications suivantes avec une explication :

- les réponses aux questions posées au §13,
- le programme de test (*bench*) écrit, en détaillant la modélisation de l'environnement et en argumentant avec précision pourquoi ce test permet d'assurer que le fonctionnement du filtre est correct,
- rapport sur la surface du filtre (#CLB, #FF, #IO) après la synthèse et après le placement et routage,
- le délai du chemin critique après le placement et routage,
- en déduire à quelle vitesse maximale le filtre pourrait fonctionner sur le FPGA choisi, après la synthèse et après le placement et routage,
- le schéma niveau portes logiques après la synthèse.
- à la fréquence de fonctionnement choisie de 50 MHz, quelle est la fréquence maximale d'échantillonnage du système ? Répondre à cette question de deux façons différentes : calcul théorique et mesure,
- préciser la fréquence de coupure du filtre à -3db (résultat de mesure).