



Laboratoire de Systèmes Logiques

Outil graphique de conception de réseaux de molécules

Edouard FORLER

Assistant
Gianluca TEMPESTI

Projet de 8^{ème} semestre
Professeur Daniel MANGE

Lausanne, juin 1999

Chapitre 1

Introduction

1.1 Motivations

Alors même que les circuits à haute intégration deviennent de plus en plus petits et de plus en plus complexes, ceux-ci sont de plus en plus sensibles à l'apparition de fautes de fabrication et de fonctionnement. Comment résoudre ces problèmes ?

C'est ainsi qu'est né au Laboratoire de Systèmes Logiques de l'EPFL le projet *Embryonics* dont le but est de développer un nouveau type de FPGA à haute complexité, dont les concepts sont inspirés de mécanismes que la Nature utilise, elle, depuis toujours.

La première phase d'*Embryonics* a ainsi permis l'élaboration de deux circuits doués d'auto-réplication et d'auto-réparation, le "biodule" 601 (MicTree[®]) et la "molécule" 603 (MuxTree[®]).

La deuxième phase du projet *Embryonics* consiste à synthétiser le biodule MicTree[®] à l'aide des molécules MuxTree[®]. C'est dans ce cadre que se situe ce travail de semestre, qui vise à réaliser un outil graphique complet de développement des réseaux MuxTree[®] de manière à faciliter la synthèse du MicTree[®] et d'autres cellules.

Le biodule MicTree[®] une fois construit, il devrait être possible, dans une troisième phase, de mettre au point de véritables "organismes" composés de MicTrees[®], et également doués d'auto-réplication et d'auto-réparation. Le premier de ces organismes sera la Biowatch 2001, une montre complète et extrêmement tolérante aux pannes de toutes sortes.

1.2 Cadre administratif

Ce projet est réalisé lors du 8^{ème} semestre (été 1999) au Laboratoire de Systèmes Logiques du Département d'Informatique de l'EPFL, sous la conduite du professeur Daniel Mange, responsable du développement *Embryonics*. L'assistant responsable du suivi du projet est Gianluca Tempesti.

1.3 Remerciements

Je remercie André Stauffer pour son aide dans l'analyse du logiciel FPGA Editor sur plate-forme Macintosh, et pour m'avoir fourni quelques configurations de réseaux de molécules nécessaires aux tests du logiciel Mux_{Designer}.

Chapitre 2

Concepts

Quatre concepts importants entrent en jeu :

- Une interface graphique ergonomique. Il est souhaité que la prise en main du logiciel soit aussi aisée que possible pour quelqu'un qui connaît la famille MuxTree® ;
- Un éditeur de réseaux MuxTree®. On pourra reprendre les concepts d'édition qui ont été développés dans un précédent logiciel appelé FPGA Editor ;
- Un simulateur des réseaux ainsi créés. Ce simulateur existe à l'état de moteur ; il a été développé lors d'un projet du semestre d'hiver 1998-99 ;
- Enfin, il serait agréable de pouvoir exploiter la production du logiciel pour configurer directement de véritables réseaux, par le transfert des informations nécessaires dans une ROM.

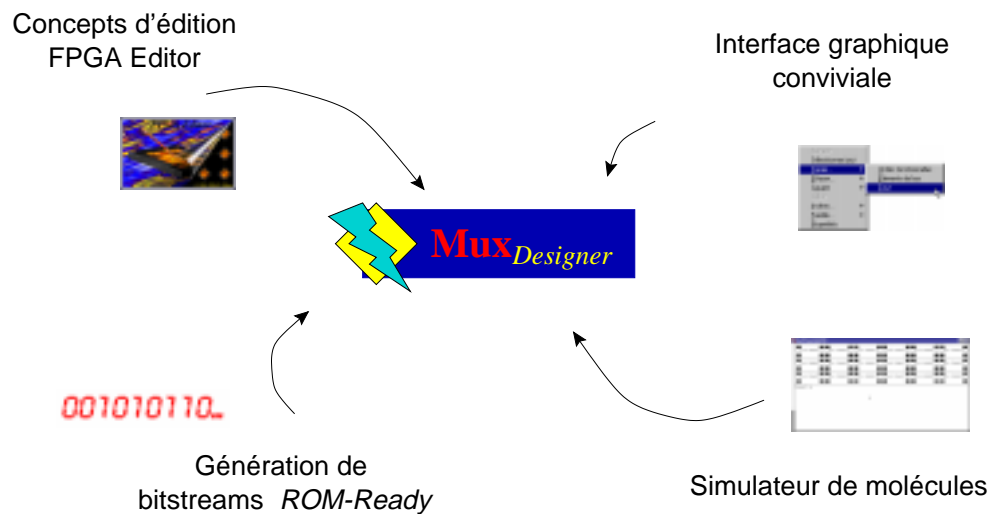


FIG. 2.1: La structure de MuxDesigner

C'est à partir de ces quatre critères de bases qu'est né le logiciel MuxDesigner. Nous ne reviendrons pas ici sur les caractéristiques du logiciel FPGA Editor et du moteur de simulation MuxTree®, disponibles au lecteur dans [1] et [2].

Pour le simulateur également, un effort a été fait sur la simplicité d'utilisation. Les propriétés d'une molécule sont accessibles d'un simple clic, et des entrées/sorties complexes peuvent être ajoutées en quelques clics et glissés de souris. La gestion des paramètres de simulation et des points d'arrêt est très visuelle.

2.2 Génération des bitstreams

Le but de notre logiciel n'est pas seulement de simplifier la conception des réseaux ; il s'agit également d'automatiser un certain nombre d'étapes, réalisées jusqu'ici manuellement. Ces étapes sont particulièrement fastidieuses et ralentissent notablement l'arrivée à maturité.

En particulier, la génération des deux séquences de configuration du réseaux, qui doivent être par la suite embarquées dans une ROM, est une étape complexe. Plusieurs manipulations doivent être effectuées afin de placer les bits des séquences dans un ordre particulier.

Cette étape sera dorénavant prise en charge par le logiciel *Mux_{Designer}*.

2.3 Autres particularités

Quelques fonctionnalités supplémentaires ont été rajoutées, comme la possibilité d'exporter les données dans un format ASCII configurable (concept repris de FPGA Editor). Naturellement, les fichiers générés par FPGA Editor sont importables dans *Mux_{Designer}*. Un certain soin a également été apporté au dessin des schémas dans l'éditeur ainsi qu'à leur impression.

D'une manière générale, l'interface du logiciel est très riche, et les possibilités nombreuses, tant au niveau de l'édition que de la simulation.

Chapitre 3

Réalisation pratique

3.1 Outils de développement

Par souci de simplicité, nous avons choisi de développer *Mux_{Designer}* sous Microsoft Windows[®] dont l'interface graphique est très aboutie et flexible.

Puisque le simulateur conçu au semestre d'hiver a été écrit en GNU C++, il est naturel de continuer le travail avec C++ sous Windows[®]. Nous avons donc recherché un outil en C++ capable de réaliser des prouesses en matière d'interface graphique. Notre choix s'est finalement porté sur C++ Builder 2.0[®] de Borland, qui est une adaptation du très populaire Delphi[®] du même éditeur au langage C++. C++ Builder[®] fait donc partie de la panoplie d'outils *RAD* (*Rapid Application Development*) de Borland et a comme avantage sur son principal concurrent Microsoft Visual C++[®] d'être beaucoup moins lourd pour la programmation de l'interface graphique. Précisons encore que l'application générée fonctionne indifféremment sous Microsoft Windows 95/98[®] et Microsoft Windows NT[®].

3.2 Mécanismes de l'interface graphique

Dans l'ensemble, ce sont des classes C++ Builder[®] standard qui ont été utilisées pour composer l'interface graphique. Deux exceptions sont à mentionner :

- le conteneur des cellules pendant l'édition et des molécules pendant la simulation est une nouvelle classe appelée **TBioGrid**, dérivée de la classe standard **TImage**. Cette nouvelle classe s'occupe de gérer l'édition dite "intelligente", du dessin des schémas et des molécules et réalise l'interfaçage avec le moteur de simulation ;
- **TBioGrid** fait appel à une seconde classe inédite, **TBioPanel**, dérivée de la classe standard **TPanel**. Cette classe fournit le panneau de gestion des entrées/sorties dans le simulateur.

Ces deux classes graphiques n'ont pas été incluses dans la bibliothèque des composants C++ Builder[®] : La disposition des objets qu'elles contiennent a été décrite "à la main" et non générée par l'outil de développement. Elles ne sont par conséquent pas visibles dans la fiche principale de l'application.

La fiche principale décrit la classe **TMainForm** qui est celle qui encapsule tous les éléments de l'application et qui gère les menus au sommet de la fenêtre. Cette classe fait appel, outre les deux classes mentionnées plus haut, à deux classes "maison" supplémentaires :

- La classe `TBioCell` contient toutes les informations de configuration d'une cellule d'édition et à fortiori, les valeurs de configuration de la molécule correspondante ;
- La classe `TBioList` encapsule une liste d'éléments de type `TBioCell`.

3.3 Portage des concepts existants

3.3.1 Moteur de simulation

Ce moteur [2] a été écrit sur station Sun à l'aide de GNU C++. De plus, aucun concept avancé (Strings, Vectors, templates, etc.) n'avait été utilisé, afin de garantir un maximum de portabilité. Cette approche se révèle être excellente ; l'inclusion des différentes classes composant le moteur s'est faite très facilement. Seuls quelques problèmes concernant des identificateurs réservés sont à signaler (en particulier, la classe `TObject` existe déjà, et il a fallu renommer celle du simulateur en `TBioObject`).

3.3.2 FPGA Editor

L'adaptation de FPGA Editor [1] au monde PC (l'outil initial fonctionne sur Apple MacIntosh[®]) a posé beaucoup plus de problèmes. En effet, aucune documentation n'est disponible et le code source a été perdu. Il a donc fallu analyser le comportement du logiciel afin de le reproduire. On peut dire qu'il a été presque entièrement réécrit pour PC ; Beaucoup de soin a été mis dans la reproduction de l'interface d'édition et dans le tracé des schémas. La majeure partie du temps de développement a été consacré au portage de FPGA Editor.

Chapitre 4

Manuel de référence

4.1 Présentation de l'interface utilisateur

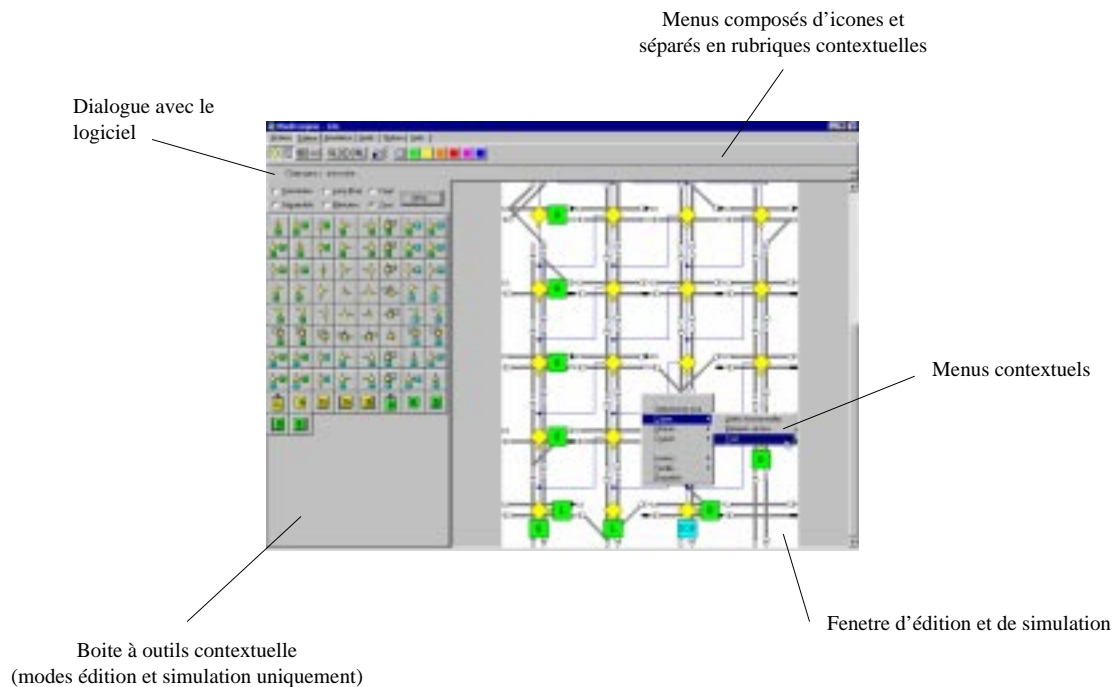


FIG. 4.1: Une vue de l'interface utilisateur

L'interface comporte quatre parties. Au sommet de la fenêtre, on retrouve les menus généraux. Ceux-ci se présentent sous la forme d'une barre d'outils à onglets. Chaque onglet donne l'accès à un contexte particulier de l'application : "Fichiers" pour la gestion des fichiers et l'impression, "Edition" pour la conception des réseaux, "Simulation" pour leur test ; "Outils" contient les fonctions d'export, "Options" permet de configurer le logiciel à sa convenance ; on trouve pour finir "Aide", dont la fonction est classique.

Chacun de ces contextes regroupe l'ensemble des outils les plus utilisés à un moment donné. Ainsi, tout est accessible directement sans devoir naviguer dans une grande arborescence de menus. Chaque option est symbolisée par une icône. En cas de doute sur leur signification,

il suffit de rester quelques secondes sur l'une d'entre elles pour que s'affiche un petit texte explicatif.

Sous les menus, se trouve une barre de message que le logiciel utilise pour communiquer avec l'utilisateur. C'est ici qu'apparaissent tous les messages d'erreur, les indications de progression et les conseils. Il est possible de faire défiler l'historique des messages grâce aux boutons situés tout à droite de la barre.

La plus grande partie de la fenêtre est occupée par le design du réseau. Selon que l'on se situe en mode "Edition" ou "Simulation", l'aspect du réseau change ; En mode "Edition", on obtient le schéma du circuit, alors qu'en mode "Simulation", c'est une représentation plus concrète des molécules qui s'affiche.

Cette partie de l'application fournit également des menus contextuels, qui contiennent les fonctions applicables à l'objet pointé.

Enfin, on trouvera, en mode "Edition" et "Simulation" uniquement, une boîte à outils sur la gauche de la fenêtre.

4.2 Liste des commandes

4.2.1 Menu *Fichiers*



Commande **Nouveau design**

Permet de créer un nouveau design. Le logiciel demande la taille initiale du réseau. La taille minimale est de 2x2. Cette taille pourra être modifiée au fur et à mesure des besoins pendant l'édition.



Commande **Ouvrir**

Permet d'ouvrir un design existant dans un fichier `.mux`. Cette commande n'accepte que les fichiers au format `MuxDesigner`.



Commande **Sauvegarder**

Permet de sauvegarder le design courant. La première fois, un nom de fichier est demandé.



Commande **Sauvegarder sous**

Permet de changer le nom du fichier du design courant.



Commande **Sauvegarder une copie sous**

Permet de créer une copie du design courant dans un autre fichier.



Commande **Version précédente**

Permet de revenir à la dernière version sauvegardée du design. Toutes les modifications réalisées entre temps sont perdues.



Commande **Importer**

Permet d'importer des données qui ne sont pas au format Mux_{Designer} pour créer un design. Pour l'instant, seul le format ASCII par défaut de FPGA Editor est reconnu.



Commande **Mise en page**

Affiche la fenêtre standard de préparation de l'impression.



Commande **Imprimer**

Lance l'impression du schéma du design entier ou de la partie sélectionnée. La mise en page est automatique, de sorte que le schéma est toujours centré et à la bonne échelle.



Commande **Quitter**

Quitte le logiciel.

4.2.2 Menu *Edition*



Commande **Afficher les unités fonctionnelles**

Si ce bouton est enfoncé, les unités fonctionnelles avec leurs connexions courtes sont affichées.



Commande **Afficher la couche bus**

Si ce bouton est enfoncé, les connexions longues et les switchblocks sont affichés.



Commande **Afficher la grille**

Si ce bouton est enfoncé, la grille qui délimite les cellules d'édition est affichée.



Commande **Afficher les coordonnées/noms**

Si ce bouton est enfoncé, les coordonnées de chaque cellule d'édition ou molécule sont affichées. Sinon, ce sont les noms des cellules ou molécules qui sont affichés.



Commande **Zoom arrière**

Ce bouton réduit la taille du réseau à l'écran.



Commande **Centrer sur**

Ce bouton fait apparaître la fenêtre de navigation dans le réseau :



FIG. 4.2: Navigation dans le réseau

Un clic dans le diagramme permet de recentrer l'édition. Il est aussi possible de se déplacer en donnant des coordonnées ou un nom de cellule.



Commande **Zoom avant**

Ce bouton agrandit la taille du réseau à l'écran.



Commande **Librairie**

Permet d'inclure un design existant dans le design courant. Après avoir donné le nom du fichier source (au format `MuxDesigner` uniquement), un clic dans la fenêtre d'édition permet de positionner le design chargé. Cette commande est utile pour construire des réseaux complexes intégrant plusieurs fonctionnalités.



Commande **Famille**

Il est possible de différencier certaines zones dans le schéma en leur attribuant des couleurs. Ainsi, pour de grands réseaux, il devient aisé de repérer des régions s'occupant de tâches différentes dans le même réseau.

Boîte à outils d'édition



FIG. 4.3: Boîte à outils d'édition

Cette boîte à outils sert à l'édition des unités fonctionnelles. Pour placer une unité dans le design, il suffit de cliquer sur le bouton correspondant puis dans la cellule où on désire placer l'unité. Les boutons radio au sommet de la boîte permettent de sélectionner une famille d'unités fonctionnelles, afin d'accélérer la recherche. Une fois l'unité placée, l'ensemble des configurations est à nouveau disponible.

Il est également possible d'ajouter une unité en spécifiant soit son code hexadécimal, soit l'arrangement de ses bits de configuration à l'aide du bouton "Héxa...", qui fait apparaître la fenêtre suivante :

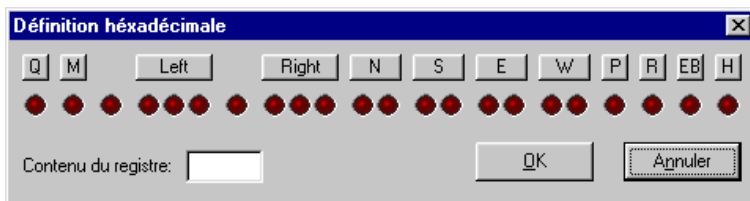


FIG. 4.4: Edition hexadécimale

Menu contextuel

En cliquant sur une cellule d'édition avec le bouton droit, un menu contextuel devient disponible. Il regroupe les outils nécessaires pour appliquer des modifications à la cellule courante ou à une sélection.

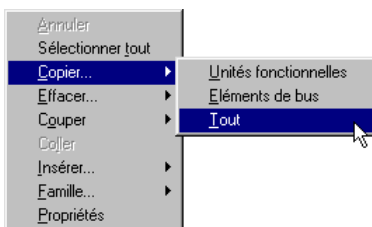


FIG. 4.5: Menu contextuel (édition)

4.2.3 Menu *Simulation*



Commande **Tout réinitialiser**

Cette commande réinitialise l'ensemble du réseau moléculaire, registres et automate cellulaire. On se retrouve juste avant le chargement des séquences de configuration.



Commande **Réinitialiser les unités fonctionnelles**

Cette commande rétablit les valeurs initiales de toutes les unités fonctionnelles séquentielles. On se retrouve juste après le chargement des séquences de configuration.



Commande **Lancer la simulation**

Démarre la simulation du réseau. Un clic sur le même bouton arrête la simulation. La simulation s'arrête également si on change de contexte (retour à l'éditeur par exemple).



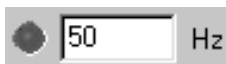
Commande **Pas à pas**

Effectue un pas de simulation.



Commande **Vitesse maximale**

La simulation a lieu à la vitesse la plus rapide possible. Il n'est plus possible de régler la fréquence de l'horloge dans ce cas. Le simulateur calcule le meilleur compromis pour répartir le temps CPU entre les calculs de simulation et l'affichage.



Commande **Fréquence de simulation**

Permet de régler la fréquence de l'horloge qui rythme le réseau pendant la simulation. Le témoin lumineux de l'horloge (à gauche) peut prendre deux couleurs : s'il est rouge, c'est le signal d'horloge rapide CCK qui est piloté. S'il est vert, c'est le signal d'horloge lente FCK qui est piloté.

Propriétés et gestion des entrées/sorties

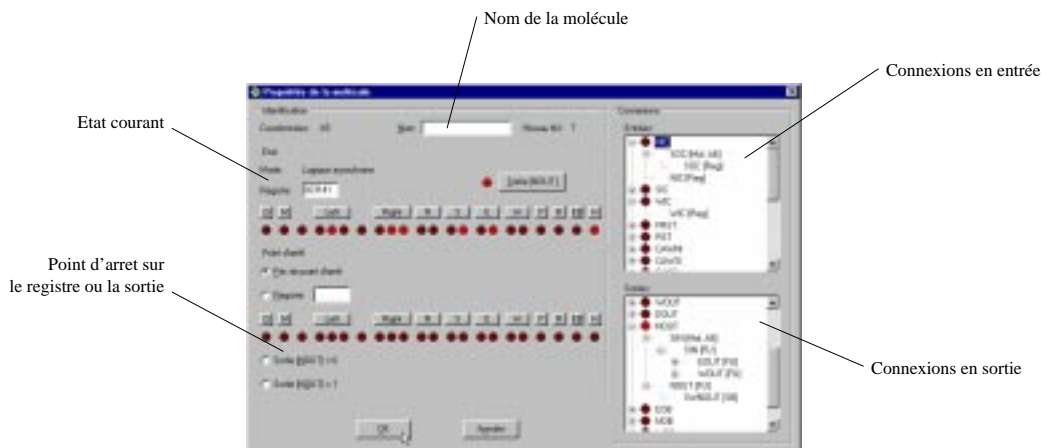


FIG. 4.6: Propriétés d'une molécule

Cette fenêtre permet de modifier le comportement d'une molécule et de placer des points d'arrêts. Elle est accessible en cliquant avec le bouton droit sur une molécule du réseau et en choisissant l'option "Propriétés".

L'autre option disponible, "Ajouter contrôle d'entrée/sortie", sert à construire une interface utilisateur permettant de manipuler certaines entrées et d'observer certaines sorties, bit à bit. Ces bits de contrôle viennent prendre place dans la fenêtre de gauche.

Il est fastidieux de réaliser des tests en manipulant un à un les bits, par exemple si on a conçu un réseau qui traite des entrées à 4, 8 ou 16 bits. Dans ce cas, il est possible, en sélectionnant des bits de contrôle déjà créés et en les ordonnant, de construire une entrée (ou une sortie) à plusieurs bits, que l'on peut piloter en entrant directement les valeurs décimales au clavier. Pour cela, il suffit de glisser les bits sélectionnés dans la liste inférieure.

4.2.4 Menu *Outils*



Commande **Exporter**

Exporte les caractéristiques du réseau (position des molécules, configuration des registres) dans un format ASCII selon les indications fournies par les **options d'exportation**.



Commande **Générer les bitstreams**

Exécute la génération des bitstreams de configuration et stocke le résultat dans un fichier ASCII. Un organisme pouvant contenir plusieurs copies du même réseau, ainsi que des molécules de rechange, il convient de spécifier sa taille, ainsi que la disposition des colonnes de molécules de rechange.

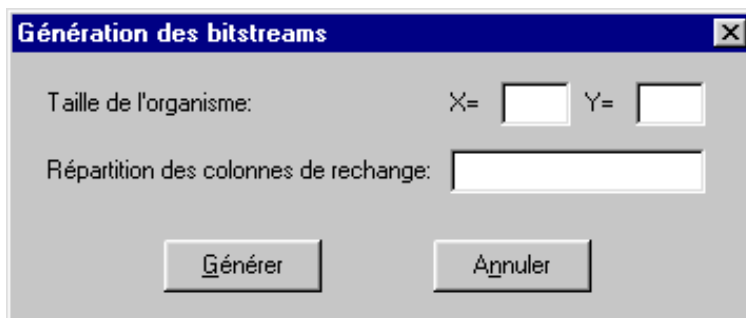


FIG. 4.7: Paramètres de génération

La disposition des colonnes de rechange est définie par une expression du type $(x-yS)^+$ ou x et y sont respectivement le nombre de colonnes actives et de colonnes de rechange alternées. Si le nombre de colonnes actives n'est pas suffisant pour contenir le design, la séquence est répétée jusqu'à l'obtention d'une largeur correcte.

4.2.5 Menu *Options*



Commande **Options d'exportation**

Permet de spécifier le format du fichier ASCII pour l'exportation. Le format par défaut est celui proposé dans FPGA Editor, mais il est possible d'en changer.

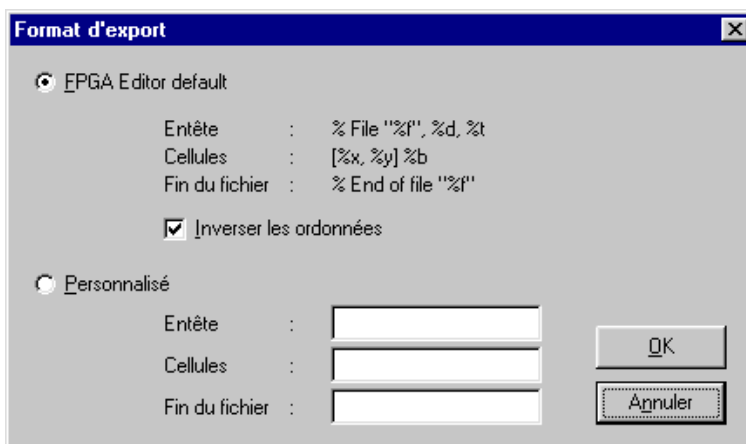


FIG. 4.8: Configuration de l'exportation

Trois parties doivent être définies :

- l'entête de fichier, qui peut contenir les symboles spéciaux **%f** (nom du design), **%d** (date d'exportation) et **%t** (heure d'exportation) ;
- la configuration des molécules, qui peut contenir les symboles spéciaux **%x** (abscisse), **%y** (ordonnée) et **%b** (valeur du registre) ;

- le terminateur, dont la syntaxe est identique à celle de l'entête.



Commande **Format des coordonnées**

Permet de spécifier la manière dont doivent s'afficher les coordonnées. Deux formats prédéfinis sont proposés : notation mathématique, (1,1) ou notation tabulaire, A1. Pour définir son propre format, il faut utiliser les symboles spéciaux %x (abscisse), %y (ordonnée), %a (abscisse alphabétique) et %b (ordonnée alphabétique).

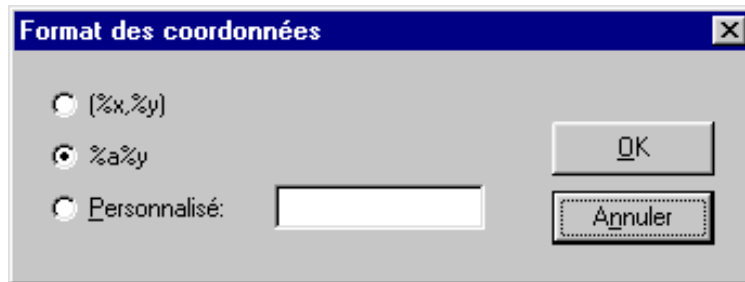


FIG. 4.9: Format des coordonnées



Commande **Options de génération des bitstreams**

Permet de sélectionner la destination et le format des bitstreams. Pour le moment, il est seulement possible de créer un fichier contenant la séquence, qui peut prendre trois formes différentes :

- séquence binaire : une suite de 0 et de 1 ;
- séquence hexadécimale : une suite de chiffres hexadécimaux ;
- mots hexadécimaux : une suite de mots de 32 bits codés en hexadécimal.

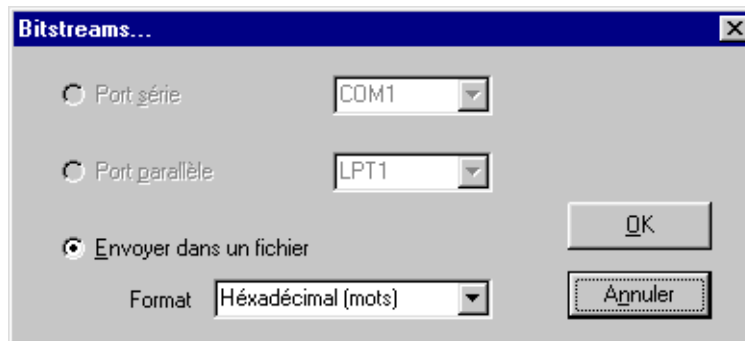


FIG. 4.10: Destination des bitstreams

4.2.6 Menu *Aide*



Commande **Aide**

Permet d'accéder à l'index de l'aide.



Commande **A propos de**

Affiche des informations concernant le logiciel.

4.3 Interaction avec l'environnement

4.3.1 Chargement automatique d'un design

`MuxDesigner` accepte un nom de fichier en paramètre. Il est possible de charger un design contenu dans un fichier `.mux` en double-cliquant dessus. Pour cela, il faut mémoriser ce type de fichier dans l'explorateur de Windows[®], en activant les options du menu "Affichage" et en choisissant l'onglet "Types de fichiers". Il faut créer un type de fichier `MuxDesigner` avec comme suffixe `.mux` et l'action associée est "ouvrir" avec `MuxDesigner.exe %1` comme commande.

4.3.2 Fichier de configuration `muxdsgn.ini`

`MuxDesigner` conserve sa configuration dans le fichier `muxdsgn.ini`. Il n'est pas conseillé de modifier le contenu de ce fichier à la main. Si le fichier n'existe pas, il est automatiquement créé avec des valeurs par défaut dans le répertoire où se trouve le programme.

Chapitre 5

Conclusion

Des résultats ont été obtenus dans les domaines suivants :

- Intégration dans un seul logiciel des outils développement autrefois séparés, parfois sur des plate-formes hétéroclites (Conception sur Apple MacIntosh[®], simulation sur station de travail Unix ou sur PC, création des séquences de configuration sur PC et sur papier...);
- Elaboration d'une interface utilisateur ergonomique, simple d'usage, qui laisse à portée de souris les outils indispensables dans un contexte précis;
- Simulation de réseaux de taille élevée (plusieurs centaines de molécules);
- Génération des séquences de configuration.

Une version beta fonctionnelle du logiciel a été développée (voir annexe A).

Le sujet étant un peu ambitieux pour un simple projet de semestre, tout n'a pu être réalisé dans les temps. Nous souhaitons cependant vivement mener à terme les travaux en cours, afin de fournir un outil permettant de poursuivre les efforts menés dans le domaine de l'embryonique au Laboratoire de Systèmes Logiques de l'EPFL.

Annexe A

Eléments à compléter

La version courante du logiciel porte le numéro 0.93. C'est une version beta, réalisée en guise de démonstrateur durant le semestre. De nombreuses fonctionnalités n'ont pas encore été développées ou seulement partiellement :

- Dans l'onglet "Fichiers" :
 - Toutes les options sont fonctionnelles à 100%. L'importation de fichiers ne prend en compte que les fichiers ASCII générés par FPGA Editor. Le code source de ce dernier programme étant indisponible, il n'est pas possible pour le moment d'exploiter le format de fichier interne à FPGA Editor.
- Dans l'onglet "Edition" :
 - La fonction de centrage (icône "viseur") n'est pas implémentée ;
 - Toutes les fonctions d'édition sont manquantes : librairie, familles, insertion d'éléments, des bus, de colonnes supplémentaires, programmation hexadécimale, propriétés ;
 - La nouvelle version du MuxTree[®] n'est pas encore exploitée (colonnes de mémoire).
- Dans l'onglet "Simulation" :
 - La réinitialisation des unités fonctionnelles seules n'a pas été implémentée ;
 - L'option "vitesse maximale" n'a pas été implémentée : elle fait appel à un calcul de performance à la fois de la simulation et de l'affichage dont l'algorithme a été étudié ;
 - Le centrage ne fonctionne pas (fonction identique à celle de l'éditeur) ;
 - La fenêtre des propriétés est largement incomplète. Bien que les points d'arrêt aient été implémentés, l'interface ne permet pas encore de les exploiter ;
 - La gestion des entrées/sorties est incomplète (effacement, renommage, suppression, etc.)
- Dans l'onglet "Outils" :
 - La génération des bitstreams est incorrecte ; l'analyse de l'expression pour la construction de la membrane cellulaire n'est pas implémentée et l'ordre des bits de configuration des registres est faux.

- Dans l'onglet "Options" :
 - Toutes les fonctionnalités sont opérationnelles à 100%. Il n'est pas possible de sélectionner les ports série ou parallèle pour l'envoi d'un bitstream, car cela suppose l'existence d'une interface matérielle restant à développer.
- Dans l'onglet "Aide" :
 - Le fichier d'aide `muxdsgn.hlp` n'existe pas encore.
- Il serait intéressant de construire l'installateur InstallShield adéquat, pour obtenir un produit final complet.

Signalons enfin que le simulateur développé au semestre d'hiver n'a pas été testé en profondeur, par manque de configurations à lui soumettre. En particulier, la molécule en mode mémoire n'a pas été testée. Quelques bugs restent à corriger.

Nous estimons qu'un mois de travail supplémentaire à temps plein permettrait d'implémenter jusqu'à 75% des fonctions manquantes.

Annexe B

Contenu des fichiers sources

Le lecteur trouvera ci-dessous la description de chacun des fichiers sources composant le projet :

Fichier .cpp	Fichier .h	Description du contenu
603class.cpp	603class.h	Description des éléments d'un MuxTree [®]
about.cpp	about.h	Fenêtre "A propos..."
analyse.cpp	analyse.h	Outils d'analyse de chaînes de caractères
biocell.cpp	biocell.h	Classes TBioCell, TBioList, TBioGrid et TBioPanel
biodules.cpp	biodules.h	Classe TMainForm
bioproj.cpp	bioproj.h	Main de l'application
bkpts.cpp	bkpts.h	Points d'arrêts et classe élémentaire TBioObject
center.cpp	center.h	Fenêtre "Centrer sur..."
coords.cpp	coords.h	Fenêtre d'options "Coordonnées"
export.cpp	export.h	Fenêtre d'options "Exporter..."
	flipflop.h	Classe TFlipFlop
generer.cpp	generer.h	Fenêtre "Générer les bitstreams..."
hexa.cpp	hexa.h	Fenêtre d'édition hexadécimale
props.cpp	props.h	Fenêtre "Propriétés"
ioform.cpp	ioform.h	Fenêtre "Ajout d'un contrôle bit"
modifwarn.cpp	modifwarn.h	Avertissement modif. sur entrée/sortie
newdesign.cpp	newdesign.h	Fenêtre "Nouveau design"
savewarn.cpp	savewarn.h	Confirmation de sauvegarde
streamopt.cpp	streamopt.h	Fenêtre d'options "Générer les bistreams..."
tbase.cpp	tbase.h	Classes de base pour les éléments de MuxTrees [®]
tbit.cpp	tbit.h	Classes de gestion des bits d'information
tmuxnet.cpp		Classe encapsulant un réseau MuxTree [®]
	types.h	Types élémentaires pour le simulateur
wordctrl.cpp	wordctrl.h	Fenêtre "Ajout d'un contrôle mot"

Annexe C

Procédure d'installation

Trois fichiers sont nécessaires pour réaliser une installation correcte : `MuxDesigner.exe`, `crysta.ttf` et `vc135.bpl`. Les étapes d'installation sont les suivantes :

- Créer un répertoire dans lequel placer les fichiers `MuxDesigner.exe` et `vc135.bpl` ;
- Installer la fonte de caractères True Type *Crystal* à l'aide du fichier `crysta.ttf` ;
- Lancer `MuxDesigner`. Le programme va générer son fichier de configuration `muxdsgn.ini`.

Pour un fonctionnement optimum du logiciel, il est recommandé de configurer l'écran en 16 millions de couleurs et avec les petites fontes de caractères. `MuxDesigner` est capable de charger au démarrage un design fourni en paramètre. On peut donc, si on le désire, configurer Windows® de manière à ce que les fichiers portant l'extension `.mux` soient automatiquement associés à `MuxDesigner`.

Bibliographie

- [1] Pascal Felber. *BDD-oriented FPGA editor*, Technical report. Laboratoire de Systèmes Logiques, EPFL, 1994.
- [2] Edouard Forler. *Outil de simulation de réseaux de molécules*, rapport de projet de semestre. Laboratoire de Systèmes Logiques, EPFL, 1998.

Index

- MicTree[®], 1
- MuxTree[®], 1
- Rapid Application Development*, 5

- Barre de messages, 8
- Borland C++ Builder[®], 5
- Borland Delphi[®], 5

- Cellule d'édition, 3
- Classes
 - TBioCell, 6
 - TBioGrid, 5
 - TBioList, 6
 - TBioObject, 6
 - TBioPanel, 5
 - TMainForm, 5
- Commandes
 - A propos de, 16
 - Afficher la couche bus, 9
 - Afficher la grille, 9
 - Afficher les coordonnées/noms, 9
 - Afficher les unités fonctionnelles, 9
 - Aide, 16
 - Boîte à outils d'édition, 11
 - Centrer sur, 10
 - Définition hexadécimale, 11
 - Exporter, 13
 - Famille, 10
 - Format des coordonnées, 15
 - Fréquence de simulation, 12
 - Générer les bitstreams, 13
 - Importer, 9
 - Imprimer, 9
 - Lancer la simulation, 12
 - Librairie, 10
 - Menu contextuel d'édition, 11
 - Mise en page, 9
 - Nouveau design, 8
 - Options d'exportation, 14
 - Options de génération des bitstreams, 15
 - Ouvrir, 8
 - Pas à pas, 12
 - Quitter, 9
 - Réinitialiser les unités fonctionnelles, 12
 - Sauvegarder, 8
 - Sauvegarder sous, 8
 - Sauvegarder une copie sous, 8
 - Tout réinitialiser, 12
 - Version précédente, 9
 - Vitesse maximale, 12
 - Zoom arrière, 10
 - Zoom avant, 10

- Edition intelligente, 3
- Embryonics, 1

Table des figures

2.1	La structure de Mux _{Designer}	2
2.2	Un aperçu des menus de Mux _{Designer}	3
2.3	L'édition "intelligente"	3
4.1	Une vue de l'interface utilisateur	7
4.2	Fenêtre de navigation dans le réseau	10
4.3	Boîte à outils d'édition	11
4.4	Edition hexadécimale	11
4.5	Menu contextuel (édition)	12
4.6	Propriétés d'une molécule	13
4.7	Paramètres de génération	14
4.8	Configuration de l'exportation	14
4.9	Format des coordonnées	15
4.10	Destination des bitstreams	15

Table des matières

1	Introduction	1
1.1	Motivations	1
1.2	Cadre administratif	1
1.3	Remerciements	1
2	Concepts	2
2.1	L'interface graphique	3
2.2	Génération des bitstreams	4
2.3	Autres particularités	4
3	Réalisation pratique	5
3.1	Outils de développement	5
3.2	Mécanismes de l'interface graphique	5
3.3	Portage des concepts existants	6
3.3.1	Moteur de simulation	6
3.3.2	FPGA Editor	6
4	Manuel de référence	7
4.1	Présentation de l'interface utilisateur	7
4.2	Liste des commandes	8
4.2.1	Menu <i>Fichiers</i>	8
4.2.2	Menu <i>Edition</i>	9
4.2.3	Menu <i>Simulation</i>	12
4.2.4	Menu <i>Outils</i>	13
4.2.5	Menu <i>Options</i>	14
4.2.6	Menu <i>Aide</i>	16
4.3	Interaction avec l'environnement	16
4.3.1	Chargement automatique d'un design	16
4.3.2	Fichier de configuration <code>muxdsgn.ini</code>	16
5	Conclusion	17
A	Eléments à compléter	18
B	Contenu des fichiers sources	20
C	Procédure d'installation	21

Bibliographie	21
Index	22
Table des figures	24