

Laboratoire n°2

Conception d'un microprocesseur

Professeur:	Chargé de cours:	Chargée de laboratoire:
Yvon Savaria	Mickaël Fiorentino	Érika Miller-Jolicoeur
yvon.savaria@polymtl.ca	mickael.fiorentino@polymtl.ca	erika.miller-jolicoeur@polymtl.ca

Automne 2020

TABLE DES MATIÈRES

1 Introduction	2
1.1 Objectifs du laboratoire	2
1.2 Dossier de travail	2
1.3 Barème de notation	3
1.4 Pénalités de retard	4
1.5 Rapport	4
2 Directives	5
2.1 Modules	5
2.2 Core	5
2.3 Implémentation	5
2.4 Performances	6

1 INTRODUCTION

La complexité d'un circuit numérique intégré à très grande échelle, tel qu'un microprocesseur, serait impossible à gérer avec les outils de conception de bas niveau (dessin des masques) que nous avons utilisé dans le laboratoire n°1. Pour mieux gérer la complexité des circuits intégrés, tout en gardant les coûts de développement dans des limites raisonnables, les outils de synthèses associés aux langages de description architecturale comme le VHDL ou le Verilog, ainsi que les outils de placement et routage automatisés permettent de générer le dessin des masques d'un circuit à partir de sa description comportementale de haut niveau.

1.1 OBJECTIFS DU LABORATOIRE

Ce deuxième laboratoire consiste à concevoir un microprocesseur. Il a pour objectif de vous familiariser avec la conception, la simulation, la synthèse, et l'implémentation physique de systèmes numériques complexes à l'aide d'outils de conception automatisés. Vous passerez ainsi à travers le flot de conception standard, tel que schématisé à la FIGURE 1. En particulier, ce laboratoire vous permettra de :

- Concevoir un modèle de processeur avec le langage VHDL
- Réaliser des simulations comportementales (modèle VHDL), et temporelles (*netlist* post-synthèse et post-implémentation) du processeur
- Réaliser les synthèses logiques du processeur en utilisant la technologie en 45 nm du kit GPDK045
- Inclure les structures de testabilité (*Design For Test (DFT)*) au processeur et générer des vecteurs de tests
- Réaliser les placement & routage automatisés à partir de la *netlist* post-synthèse du processeur
- Évaluer la consommation d'énergie du processeur à partir de ses informations post-implémentation et de l'activité qu'il a généré en simulation temporelle

Conformément au plan de cours, vous suivrez dans un premier temps le [tutoriel numérique](#) qui vous guidera dans l'apprentissage des techniques de simulation, de synthèse, de testabilité, de placement & routage, et d'évaluation de la consommation d'énergie avec les outils *Modelsim*, *Genus*, *Modus*, *Innovus*, et *Voltus* à travers l'exemple d'un compteur BCD. Le langage VHDL sera présenté en classe. La documentation du processeur contient toutes les informations nécessaires à sa conception. En vous aidant des tutoriels et de la documentation, vous concevrez d'abord chacun des modules composant le processeur, puis vous utiliserez ces modules pour concevoir le *core*.

1.2 DOSSIER DE TRAVAIL

Le dossier de travail associé au laboratoire n°2 est disponible sur un répertoire *git*. Téléchargez ce répertoire sur votre station de travail avant de commencer :

```
% cd ~/Labs  
% git clone https://git.step.polymtl.ca/ele8304/lab2.git
```

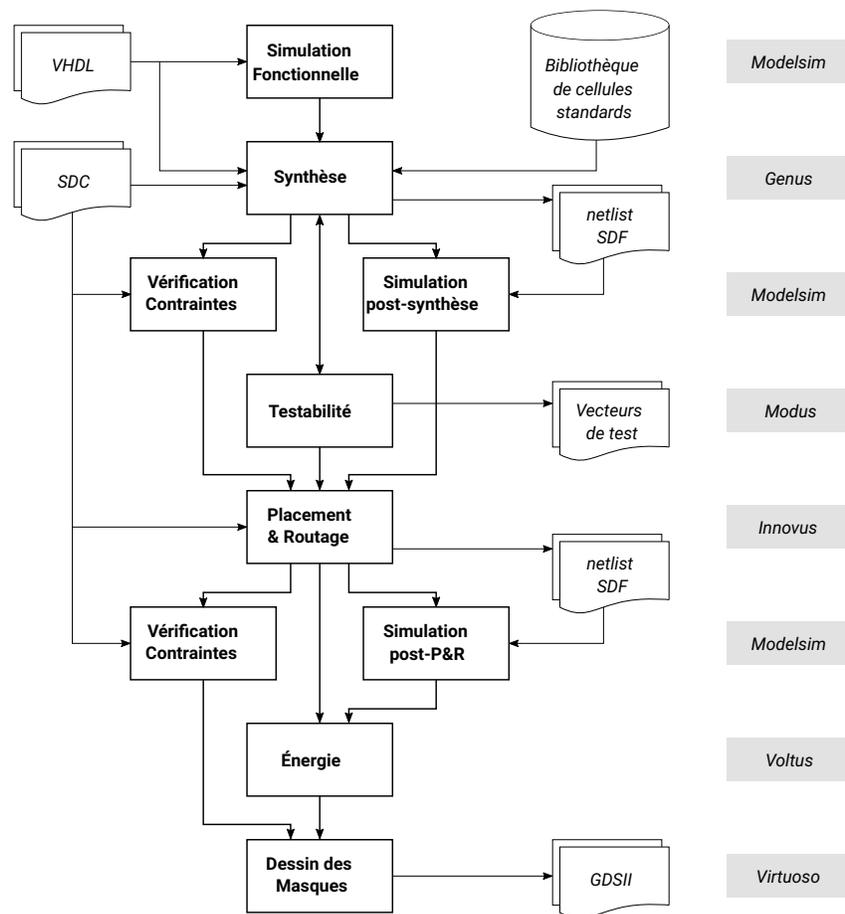


FIGURE 1: Flot de conception

Le dossier de travail est organisé de la façon suivante :

```

asm/..... # Programmes assembleur
scripts/..... # Automatisation du flot de conception
sources/..... # Code source VHDL
constraints/..... # Fichiers de contraintes
simulations/..... # Simulations comportementales et temporelles
implementation/..... # Synthèse logique, placement et routage
doc/..... # Documentation, Rapport

```

1.3 BARÈME DE NOTATION

Le TABLEAU 1 donne la répartition des points du laboratoire. 10 points sont alloués à la conception et à la simulation des modules et du *core* en VHDL. 5 points sont alloués au flot de conception (synthèse, testabilité, placement & routage, vérifications). 3 points sont alloués à l'évaluation des performances et de la consommation d'énergie du processeur, dont 1 point attribué à l'équipe qui obtiendra le meilleur ratio performance/puissance. Enfin, 2 points sont alloués à la qualité du rapport.

TABLEAU 1: Barème de notation

Modules	/5
Core	/5
Implémentation	/5
Performances	/3
Rapport	/2

1.4 PÉNALITÉS DE RETARD

La date limite de remise du laboratoire est précisée sur Moodle. Les pénalités de retard P_{retard} sont calculées en fonction du nombre h d'heures de retard tel que :

$$P_{retard} = 0.5 \times \left[1 + \left(\frac{h}{24} \right)^2 \right]$$

1.5 RAPPORT

Votre rapport doit inclure tous les éléments nécessaires à la justification de vos analyses. C'est-à-dire vos résultats de simulation, vos résultats de synthèse et d'implémentation, et tout autre élément que vous jugerez pertinent pour démontrer votre bonne compréhension du travail réalisé. Les 2 points alloués à la qualité du rapport concernent la précision du vocabulaire employé, la qualité de la présentation et des figures, ainsi que la qualité de la syntaxe et de l'orthographe. Seule la remise d'une version électronique par groupe est nécessaire. On vous demande de remettre un fichier compressé (.zip) sur Moodle avant l'échéance de remise du laboratoire contenant le dossier de travail complété de votre travail, en particulier :

- Votre rapport au format PDF.
- Vos codes sources VHDL (modules, *core*, et bancs d'essais).
- Vos scripts *tcl* pour la simulation, la synthèse, et l'implémentation.
- Vos résultats post-synthèse et post-implémentation (*netlist*, rapports de STA, DRC, LVS, rapports d'énergie, surface occupée etc.)

Remarque—*Nous souhaitons avoir vos commentaires sur les difficultés que vous avez rencontrées ainsi que le temps d'apprentissage que vous avez passé sur les outils durant la réalisation de ce laboratoire. Nous sommes particulièrement intéressés aux lacunes pouvant subsister dans la documentation.*

2 DIRECTIVES

2.1 MODULES

Dans cette partie on vous demande de concevoir les modules composant le processeur. Les explications relatives au fonctionnement des modules, leurs schémas de principe, et leurs interfaces VHDL, sont détaillés dans la documentation du processeur. En particulier, on vous demande de :

- Concevoir le modèle VHDL de tous les modules décrit dans la documentation du processeur, en respectant avec *exactitude* leurs interfaces VHDL, et de valider leur fonctionnement à l'aide d'une simulation comportementale et d'un banc d'essai. **(3 pt)**
- Réaliser la synthèse logique et la simulation post-synthèse des modules *séquentiels*. Utilisez le même banc d'essai que pour la simulation comportementale. **(2 pt)**

2.2 CORE

Dans cette partie on vous demande de concevoir le *core* du processeur en utilisant les modules réalisés précédemment. Le fonctionnement du *pipeline* est détaillé dans la documentation. En particulier, on vous demande de :

- Concevoir le modèle VHDL du *core* tel que décrit dans la documentation, en respectant avec *exactitude* son interface VHDL. **(3 pt)**
- Valider son fonctionnement à l'aide d'une simulation comportementale et d'un banc d'essai montrant l'exécution du *benchmark* Fibonacci fourni dans le dossier de travail. **(2 pt)**

Pour pouvoir exécuter un programme avec votre processeur, votre banc d'essai doit correctementinstancier la mémoire double port (`dpm.vhd`) fournie dans le dossier de travail. Notez que vous pouvez effectuer des tests complémentaires avec vos propres programmes assembleurs en utilisant le Makefile pour la compilation. Assurez vous que le fichier d'initialisation de la mémoire d'instruction (`*.hex`) reflète les modifications que vous avez apportées à votre programme après la compilation.

2.3 IMPLÉMENTATION

Dans cette partie on vous demande de réaliser l'implémentation physique du processeur en utilisant la technologie 45 nm du kit GPDK045 de Cadence. En particulier, on vous demande de :

- Réaliser la synthèse logique du processeur, en incluant la testabilité. Vérifiez que les contraintes temporelles sont respectées avec l'outil STA, et validez le fonctionnement du processeur à l'aide d'une simulation temporelle post-synthèse. Utilisez le même banc d'essai que pour la simulation comportementale, et mettez en évidence les délais dans le circuit. **(2 pt)**
- Réaliser le placement & routage du processeur. Vérifiez que les contraintes temporelles sont respectées avec l'outil STA, vérifiez l'intégrité de son dessin des masques avec les vérifications DRC et LVS, et validez le fonctionnement du processeur à l'aide d'une simulation temporelle post-placement-routage (même banc d'essai). **(2 pt)**

- Dimensionner le circuit (incluant des modifications du circuit et du flot de conception) de façon à obtenir les meilleures performances possibles. Vérifiez le respect des contraintes temporelles avec l’outil STA. **(1 pt)**

2.4 PERFORMANCES

Dans cette partie on vous demande d’évaluer les performances de votre processeur post-implémentation. C’est à dire, de mettre en parallèle le temps d’exécution du *benchmark* Fibonacci avec l’énergie consommée pendant son exécution. En particulier, on vous demande de :

- Réaliser la simulation post-implémentation du processeur en exécutant le *benchmark* Fibonacci. Utilisez le même banc d’essai que pour la simulation comportementale. Enregistrez l’activité générée par la simulation temporelle dans un fichier `vcd`. **(1 pt)**
- Déterminer la performance de votre processeur en Million d’Instructions Par Secondes (*MIPS*), en utilisant les informations fournies par votre compteur de performance. **(1 pt)**
- Évaluer la puissance moyenne P (en mW) consommée par votre processeur pendant l’exécution du programme à partir de l’activité enregistrée en simulation post-implémentation. Le ratio $MIPS/P$ est la métrique utilisée pour départager les équipes. L’équipe avec le plus haut score aura le point. **(1 pt)**