Session S4-APP1 GEGI

APP_COMBI

Guide étudiant
Logique combinatoire

Département de génie électrique et de génie informatique
Faculté de génie
Université de Sherbrooke

Hiver 2024

Note : En vue d'alléger le texte, le masculin est utilisé pour désigner les femmes et les hommes.

Document S4_APP_Comb_Guide_Etudiant_H2024.doc
Rédigé par Gérard Lachiver, ing., décembre 2001
Révisé par Réjean Fontaine, ing., novembre 2004
Révisé par Frédéric Mailhot, janvier 2006
Révisé par Réjean Fontaine, ing., janvier 2010
Révisé par Réjean Fontaine, Daniel Dalle, janvier 2014
Révisé par Réjean Fontaine, Daniel Dalle, 15 janvier 2015
Révisé par Réjean Fontaine, Amer Al-Canaan, 21 décembre 2015
Révisé par Réjean Fontaine, Daniel Dalle, décembre 2018
Révisé par Réjean Fontaine, Marc-André Tétrault, décembre 2019, Avril 2020, Janvier 2021, Décembre 2022, Avril 2022, Décembre 2022.

Table des matières

Table des figures

1. Éléments de compétences de la session S4 visés par l'unité

GEN420 – Mathématiques des circuits logiques 2 crédits

https ://www.usherbrooke.ca/admission/fiches-cours/GEN420/

Cible(s) de formation :

S4-APP1 : Modéliser et résoudre un problème de logique combinatoire et séquentielle à l'aide de représentations mathématiques de l'information discrète et par la synthèse des équations booléennes.

S4-APP2 : Modéliser l'information discrète et son évolution temporelle. Déterminer les structures de données et les algorithmes appropriés pour les mettre en œuvre.

GEN430 – Circuits logiques 2 crédits

https ://www.usherbrooke.ca/admission/fiches-cours/GEN430/

Cible(s) de formation :

S4-APP1 : Concevoir et réaliser des systèmes numériques combinatoires à partir de spécifications.

S4-APP2 : Concevoir et réaliser des systèmes numériques séquentiels à partir de spécifications.

Qualités de l'ingénieur

Les qualités de l'ingénieur visées par cette unité d'APP sont les suivantes. D'autres qualités peuvent être présentes sans être visées ou évaluées dans cette unité d'APP.

	Q01	Q02	Q03	Q04	Q05	Q06	Q07E	Q08	Q09	Q10	Q11	Q12
Touchée	X	X		X	X		X					
Évaluée		X			X							

Pour une description détaillée des qualités et leur provenance, consultez le lien BCAPG sur le site de la Faculté de génie :

Les modalités d'évaluation des compétences, les pondérations et grilles d'évaluation sont présentées à la fin de ce guide.

Énoncé de la problématique

Détecteur de proximité

Vous venez de vous découvrir une passion pour la musique et vous voulez y allier vos connaissances en génie pour compléter la conception d'un détecteur de proximité capable de produire des effets spéciaux dans votre prochain show rock. Vous voulez moduler vos compositions musicales en fonction de la distance entre votre main et un détecteur de proximité qui génère un signal analogique entre 0 et 3,3 V proportionnellement à cette distance. Pour cet APP, vous allez vous attarder à la représentation des nombres et au traitement des données en vue d'une transmission sans fil. Pour obtenir la flexibilité d'avoir un grand nombre d'entrées/sorties, vous avez opté pour un ZYNQ-7000, une famille de matrice de portes programmables (FPGA) de la compagnie *Xilinx*. Ce FPGA est soudé sur une carte électronique appelée *Zybo-Z7* provenant de la compagnie *Digilent*. Vous avez également recyclé un *ancien* circuit électronique capable d'effectuer une conversion analogique à numérique sur 12 bits à l'aide d'une série de comparateurs. La conversion analogique-numérique produit un code de type thermométrique où le nombre de « *1* » apparaît successivement depuis les LSB vers les MSB proportionnellement à l'amplitude du signal à la manière d'un thermomètre (bus ADC(11 :0) de la figure 1). Vous vous fixez comme mandat de transformer le code thermométrique pour afficher 1) la parité sur une DEL, 2) sa valeur numérique sur des afficheurs 7 segments sous différentes notations, et 3) l'amplitude du code thermométrique sur une série de DEL.

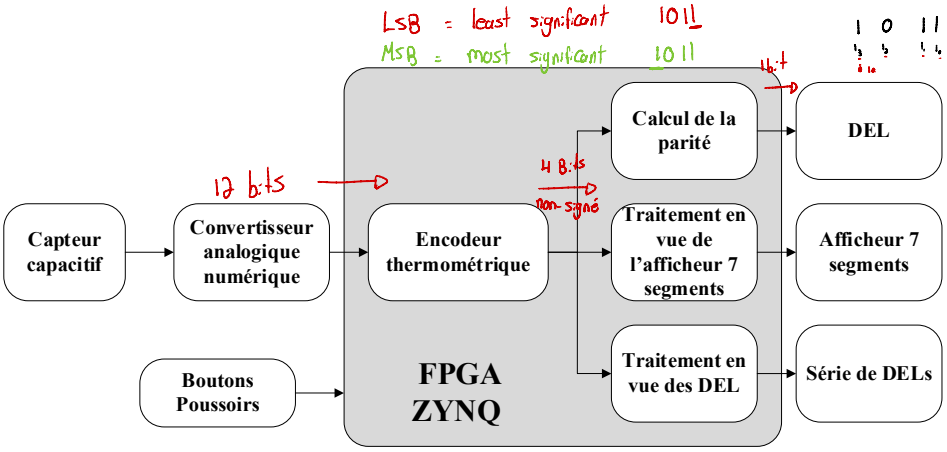


Figure 1 Schéma global du système

Vous avez commencé à produire une spécification fonctionnelle de haut niveau décrivant l'architecture électronique que vous allez utiliser (figure 1). Vous comprenez tout de suite qu'une chaîne de conversion des données doit être réalisée pour produire les différents affichages. Entre autres, vous devez concevoir un *encodeur thermométrique* capable de transformer la sortie du convertisseur analogique numérique en une représentation binaire non signée sur 4 bits propagée par le bus ADCbin(3 :0) (figure 1). En tant qu'ingénieur astucieux vous ne demeurerez certainement pas indifférent aux indifférents et simplifierez adéquatement, dans ce module, les équations logiques par l'algèbre de Boole et les tableaux de Karnaugh. Lorsque votre conception de cet encodeur thermométrique sera terminée, vous devrez calculer sa fréquence maximale d'opération. Outre l'encodeur thermométrique, vous devrez également développer trois autres modules soit le *Calcul de la parité*, le *Traitement en vue de l'afficheur 7 segments* et le *Traitement en vue des DEL*.

Le résultat du *Calcul de la parité* sur les données du bus ADCbin(3 :0) sera affiché sur la DEL *D2* de votre *ancienne* carte électronique ainsi que sur une DEL de la carte *Zybo*. Si le bouton poussoir *S1*, localisé sur votre *ancienne* carte, est pressé, la DEL *D2* et la DEL de la carte *Zybo* indiqueront une parité paire alors que si *S1* n'est pas pressé, la parité impaire sera affichée. Les DEL doivent être allumées si la

parité vaut 1 sinon elle est éteinte. Vous allez devoir ajuster votre code en fonction du nombre d'inverseurs dans la chaine et de la polarité de la DEL *D2* qui est en tirage (*pull-up* en anglais).

Le module de *Traitement en fonction de l'afficheur 7 segments* permet de transformer les quatre bits non signés ADCbin(3 :0) en diverses représentations : hexadécimale, binaire codé décimal positif et négatif. Des boutons poussoirs, localisés sur votre carte *ZYBO*, permettront de sélectionner la représentation désirée sur les afficheurs à 7 segments. La sélection du type de représentation à afficher s'effectuera par un circuit combinatoire de type multiplexeur.

Pour vous assurer que votre encodeur thermométrique fonctionne correctement, vous avez décidé de développer un module de *Traitement en vue des DEL*. Ce module transforme le signal ADCbin(3 :0) en une représentation de type one hot sur une série de 8 DEL identifiées LD(7 :0) alignées sur un module type PMOD de la compagnie *Digilent*. Le comportement en sortie sera d'allumer une DEL à la fois un peu à la manière d'un curseur (une seule DEL s'allume à la fois). Vous vous apercevez rapidement qu'il faut compresser les 12 bits sur les 8 DEL disponibles et vous avez décidé de réaliser une fonction de multiplication par 2/3 à l'aide d'additionneurs 1 bit que vous concevrez dans votre laboratoire. Pour cet additionneur *n* bit, vous cascaderez plusieurs additionneurs 1 bit avec un mécanisme de propagation de la retenue pour réaliser les calculs sur la plage dynamique désirée. Vous devrez faire attention à sélectionner le nombre minimal de bits pour vous assurer qu'il n'y ait pas de débordement de la plage dynamique. Dans le module de *Traitement en vue des DEL*, vous avez également statué que le LSB correspond au LD(0). Pour réaliser cette fonction, l'utilisation d'un décodeur 3 à 8 s'avère très appropriée.

Évidemment, le fait de recycler un *ancien* circuit électronique exige de faire des vérifications au niveau de la compatibilité des entrées/sorties et du *fanout* (ou *sortance* en français), spécialement entre le FPGA et DEL D2 de votre *ancien* circuit électronique qui ne semble pas s'allumer même si un « 1 » logique est généré par le FPGA ☹. Il faut trouver pourquoi et l'indiquer dans le rapport.

L'étude d'un système numérique comme celui impliqué dans le *Détecteur de proximité* nécessite la rédaction de deux types de spécification : les spécifications fonctionnelles et les spécifications de conception. La première décrit les fonctions que doit réaliser le circuit alors que la seconde décrit comment il sera réalisé. Le circuit se base sur une approche hiérarchique modulaire pour décrire les spécifications fonctionnelles. Cette approche est bien adaptée aux systèmes numériques réalisés en logique combinatoire. Il ne vous reste qu'à compléter les spécifications de conception et réaliser les circuits.

Pour passer de la spécification fonctionnelle à la réalisation physique du circuit, vous savez qu'il faut d'abord écrire les spécifications de conception du module. Pour cela, il faut déterminer les entrées et les sorties d'un module puis d'en comprendre le fonctionnement. L'annexe du guide étudiant est très utile à ce niveau. Une fois les spécifications établies, il faut déterminer les équations logiques capables de créer la fonction attendue. Cette deuxième étape peut se faire de deux façons. La première consiste à mettre en équation la fonction souhaitée à l'aide des théorèmes de l'algèbre de Boole et des tableaux de Karnaugh, puis à traduire les expressions logiques obtenues en un schéma faisant appel à des composants simples ou complexes (c.-à-d., portes logiques, décodeurs, multiplexeurs, circuits logiques standards, circuits logiques programmables). L'autre alternative met en œuvre un langage de description du matériel HDL (*Hardware Description Language*) et demande d'avoir accès à un environnement de conception assistée par ordinateur.

Vous allez résoudre ce problème en utilisant la description VHDL, selon les spécifications fournies dans l'annexe du guide étudiant. On y retrouve la définition des interfaces (entrées et sorties) pour chaque

module et leur comportement attendu (spécification). L'annexe vous introduit également à une réflexion sur la création des tests unitaires et d'intégration. Cependant, la liste de tests est incomplète. Il y manque les tests du module thermo2bin ainsi que ceux du système global ; vous devrez donc compléter cette liste de tests. À l'aide de cette dernière, vous validerez ensuite 1) chacun des modules individuellement par une simulation appropriée et 2) le circuit complet avant son implantation dans le FPGA. Nous vous conseillons de vous garder au moins une demi-journée pour l'intégration système des modules et l'implantation sur la carte FPGA. L'implantation exige de comprendre certains concepts notamment aux technologies (CMOS ou TTL par exemple) sur vos cartes et aux considérations de compatibilité des signaux.

Votre *ancienne* carte électronique dispose d'un potentiomètre permettant de générer une tension équivalente à celle du capteur capacitif. Bien que ce potentiomètre soit difficile à opérer car il est oxydé, vous pourrez l'utiliser pour effectuer les tests fonctionnels à la maison. Un capteur capacitif sera disponible lors de la validation de votre circuit.

① déterminer entrées/sorties
② déterminé equations logique
③ Complété tests
④ Implémentation sur FPGA

2. Connaissances nouvelles à acquérir par la résolution de cette problématique

Connaissances déclaratives : ***Quoi***

- Différence entre un système numérique et un système analogique
- Spécifications haut niveau et bas niveau d'un système numérique
- Représenter et modéliser de l'information discrète
- Type de système numérique : combinatoire
- Fonctions logiques de base et leur représentation graphique
- Utilisation de l'algèbre booléenne pour écrire une expression logique
- Mise en équation sous la forme SOP et POS
- Transformation des expressions logiques
- Description et analyse d'un réseau de portes logiques
- Caractéristiques électriques et temporelles des circuits intégrés combinatoires
- Représentation des variables binaires au niveau physique
- Calcul des temps de propagation dans un réseau de portes logiques
- Structure de base des portes logiques et leur opération
- Conception de circuits combinatoires à partir d'une description fonctionnelle
- Circuits combinatoires standards et programmables
- Circuits arithmétiques simples

Connaissances procédurales : ***Comment***

- Écrire les équations logiques d'un circuit combinatoire
- Utiliser des fonctions logiques combinatoires standards
- Éditer un schéma logique dans un environnement de conception assistée par ordinateur
- Formuler des fonctions logiques sous forme de schémas
- Mettre en œuvre des fonctions logiques dans un environnement de conception assistée par ordinateur
- Concevoir des fonctions logiques combinatoires à l'aide d'un langage de description (VHDL).
- Valider le comportement logique d'un circuit combinatoire sur un simulateur logique
- Réaliser un circuit logique combinatoire sur une matrice de portes logiques programmables
- Valider le fonctionnement d'un circuit logique combinatoire sur une matrice de portes logiques programmables
- Utiliser des appareils de mesure de base pour mettre au point un circuit logique combinatoire

Connaissances conditionnelles : ***Quand***

- Quand utiliser la logique combinatoire
- Quand simplifier une expression logique
- Quand utiliser des fonctions logiques standards
- Quand considérer les délais de propagation dans les circuits
- Choisir les ressources de conception assistée par ordinateur qui sont adéquates pour réaliser un circuit logique.

3. **Références et guide de lecture**

3.1. Références essentielles à consulter

Volume obligatoire : John F. Wakerly Digital Design, Principles and Practices, 4th edition, Prentice Hall, 2006, 895 p. ; disponible en version Kindle/électronique (visionneur logiciel gratuit Windows/Apple/Android)

Extraits disponibles sur la page de l'APP

Séquence d'étude suggérée en vue de compléter les procéduraux :

- Pour l'atelier Xilinx
  - Visionner la capsule vidéo Youtube disponible dans la section *Atelier* sur site WEB
  - Visionner les capsules vidéo *Séminaire 1 à 10, concept de base en VHDL*.
  - Visionner les capsules thématiques *Atelier 1, 2, 3, 4 et 8*.
- Pour le premier procédural :
  - Chap. 1 (lecture optionnelle mais intéressante pour s'introduire au sujet) au complet (24 p)
  - Chap. 2 paragraphes 2.1 à 2.5 (sauf 2.5.5 à 2.5.7), 2.6, 2.10, 2.11. (24 p)
  - Chap. 4 paragraphes 4.1 à 4.3 (40 p)
  - L'annexe du guide étudiant portant sur les XOR
- Pour le premier laboratoire
  - Revisionner les capsules vidéo *Séminaire 1 à 10, concept de base en VHDL*.
  - Revisionner les capsules thématiques *Atelier 1, 2, 3, 4 et 8*.
  - Sections du chapitre 2 (Wakerly) décrites pour le 1er procédural, 6.10.1 et 6.10.2

Ces lectures détaillent progressivement la représentation physique de l'information. Elles sont complétées avec les encodeurs/décodeurs et multiplexeurs qui sont des composants importants des circuits logiques.

- Pour le second procédural :
  - L'annexe du guide étudiant portant sur les notions d'électronique. Le chapitre 3 complète les lectures de l'annexe du guide étudiant. Si vous avez bien compris l'annexe du guide étudiant, lire rapidement le chapitre 3. Cependant, il faut absolument lire la section 3.6.2 et continuer au chapitre 4. Voici les sections à regarder dans le chapitre 3 si vous désirez un complétement à l'annexe du guide étudiant : paragraphes 3.1 à 3.3.3, 3.4 à 3.5.1, 3.5.4 à 3.6.4, 3.7.1 à 3.7.3, 3.7.5, 3.8.1 à 3.8.3, 3.9 à 3.9.1, 3.10.3 à 3.10.8 (60 p).
  - Chap. 4 paragraphes 4.4 (5 p)
  - Chap. 6 paragraphes 6, 6.1, 6.2, 6.4.1 à 6.4.4, 6.4.8, 6.5, 6.5.1, 6.5.2, 6.6, 6.6.1, 6.6.2, 6.7, 6.7.1 à 6.7.3, 6.8, 6.8.1 à 6.8.4, 6.9, 6.9.1 à 6.9.4, 6.10, 6.10.1 à 6.10.7, 6.11.1 (74 p)

Lecture au besoin en fonction de vos connaissances en VHDL suite au séminaire. La lecture sur les codes VHDL est principalement localisée dans le chapitre 5 et disséminée dans le chapitre 6 pour chacun des composants complexes.

ℭ Pour maîtriser les concepts du présent APP, il est essentiel de les mettre en pratique en résolvant quelques-uns des problèmes présentés à la fin de chacun des chapitres.

Outre les exercices que nous allons effectuer en pratique procédurale, nous vous suggérons particulièrement la série d'exercices suivants dans Wakerly (voir sur le site WEB) :

- Chap. 2 : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 16 ;
- Chap. 4 : 6, 7, 8, 14, 15, 18, 19 ;
- Chap. 6 : 20, 21.

Note importante : Nous supportons seulement la version professionnelle de Vivado disponible sur les serveurs du département. Les versions installées sur vos propres ordinateurs sont sous votre responsabilité.

3.2. Autres références utiles :

- Site de la compagnie Xilinx : www.xilinx.com
- Site web du fabricant de composants logiques : Texas Instruments. À partir de ce site, sélectionner une famille de circuits, par exemple : ALS ou FCT
  http ://focus.ti.com/docs/logic/logichomepage.jhtml
- Logiciel gratuit de création de chronogrammes : https://wavedrom.com/

Notes de lecture du livre de référence

note 1. Vous allez probablement vous buter sur plusieurs nouveaux termes techniques anglophones, ne vous découragez pas. Notez-les, il nous fera plaisir de vous aider à les comprendre durant les périodes de laboratoire. Cependant, il est essentiel de bien maîtriser ce vocabulaire, car il est à la base de toutes les activités techniques en génie électrique et en génie informatique. Attention aux faux amis. On traduit *digital watch* par *montre numérique* et non par *montre digitale* et on traduit *silicon* par *silicium*, exemple : un circuit intégré au silicium (le silicone c'est le *caulking* qu'on met sur le bord du *sink* !). Mais attention avec l'utilisation du terme numérique. Il peut référer à des circuits qui traitent des signaux binaires (ce que nous étudierons), mais aussi à des signaux analogiques discrétisés (passés dans un convertisseur analogique à numérique) comme en traitement numérique du signal. Dans le premier cas on dit souvent, circuits et signaux logiques.

note 2. Les transistors constituent la base de la construction des circuits intégrés. Jadis, les ingénieurs d'AMD utilisaient environ cinq milliards de transistors pour fabriquer un processeur de la Xbox One ! Le tout sur seulement 363 mm². Pensez que les temps de commutation de ces transistors sont inférieurs à la nanoseconde (10^{-9} s).

note 3. Tout le développement des circuits logiques repose sur les théorèmes et les postulats de l'algèbre de commutation qui est un sous-ensemble d'une algèbre inventée par G. Boole à la fin du 19^{e} siècle.

note 4. Faites attention à la priorité des opérateurs lorsque vous calculez le complément d'une expression ou que vous appliquez le théorème de De Morgan. Au début, utilisez des parenthèses et ne sautez pas les étapes dans le développement de la solution. Ne confondez pas dualité et complémentation, c'est une erreur fréquente.

note 5. Sachez simplifier une expression logique rapidement à l'aide d'un tableau de Karnaugh. Sur quel théorème repose cette technique graphique de simplification ? Notez bien le codage des lignes et des colonnes. On utilise rarement Karnaugh s'il y a plus de 4 variables, on utilise dans ce cas des méthodes algorithmiques de simplification.

note 6. Le concept de forme canonique est fondamental, il est à la base de la réalisation des circuits et en particulier les matrices de circuits programmables.

note 7. Familiarisez-vous avec les fonctions logiques combinatoires de base (décodeurs, multiplexeurs, OU exclusif, comparateurs, additionneurs et soustracteurs). Utilisez-les comme des blocs de construction pour générer des fonctions plus complexes. Ces fonctions élémentaires se retrouvent dans toutes les librairies de logiciels de conception assistée par ordinateur. Les buffers 3-états sont couverts dans la section électronique de l'annexe du guide étudiant.

4. Activités liées à la problématique

Semaine 1 :

- 1re rencontre de tutorat
- Étude personnelle et exercices
- Formation à la pratique procédurale 1
- Atelier de formation sur l'environnement de développement Xilinx
- Formation à la pratique en laboratoire
- Rencontre collaborative pour la résolution de la problématique
- Formation à la pratique procédurale 2

Semaine 2 :

- Remise individuelle du devoir (voir horaire).
- Validation pratique de la solution en laboratoire selon l'horaire sur le site WEB en groupe collaboratif de 2 et téléversement du projet Xilinx sur le site WEB. Notez que la validation sera effectuée par le corps professoral et compte pour l'évaluation des compétences.
- Remise du rapport d'APP **par équipe de 2** selon l'horaire sur le site WEB.
- 2e rencontre de tutorat
- Évaluation formative
- Consultation facultative
- Évaluation sommative

5. Livrables

Il y a 3 livrables dans l'APP : Un devoir, une validation et un rapport. Tout retard sur la remise d'un livrable entraîne une pénalité de 20 % par jour.

5.1. Devoir à remettre le lundi suivant le 2eme procédural avant 13h00

Le devoir doit être en format PDF, soumis avec la procédure de téléversement sur la page WEB de l'APP. Ce document est INDIVIDUEL. Utilisez la même page titre qu'en S1. Le nom du fichier doit être CIP.pdf.

Question 1.

Soit la table de conversion binaire pour un afficheur 7 segments, similaire à celui utilisé dans la problématique (Table I), servant à afficher un symbole hexadécimal (0_{16} à F_{16}) correspondant à la valeur binaire $D_3..D_0$.

Table I

D_3	D_2	D_1	D_0	A	B	C	D	E	F	G
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1

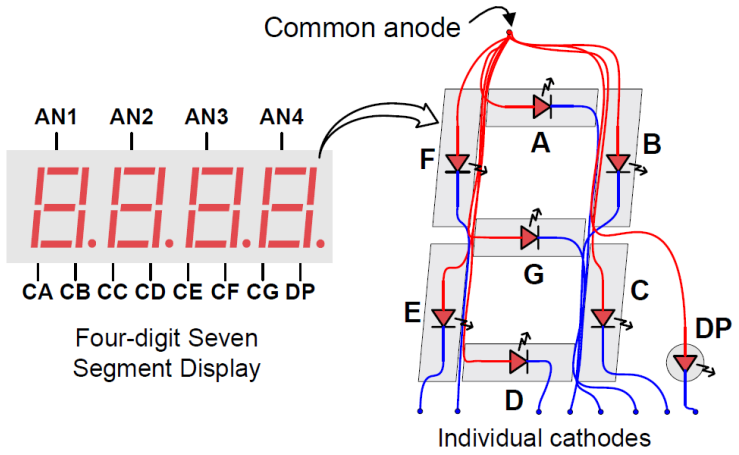


Figure 2. Afficheur 7 segments

On désire obtenir les équations simplifiées minimales nécessaires pour afficher un code BCD sur l'afficheur à 7 segments. Pour le but de cet exercice, on ne va s'intéresser qu'au segment « E » de l'afficheur 7 segments car la procédure pour les autres segments est identique.

Question 1: Démontrez que la solution la plus simple du segment « E » pour afficher un code BCD est la suivante :

$E = D_1D_0' + D_2'D_0'$

Expliquez votre cheminement complet à l'aide d'un tableau de Karnaugh réalisé sur un support électronique (ex : pdf).

Question 2 : Est-ce que l'égalité suivante est vraie ? Démontrez-le par l'algèbre de Boole.

$$D_0'D_1' + D_0'D_2 + D_1D_3 + D_2' D_3 + D_1'D_2D_3' = D_0'(D_1D_2')' + D_3 \oplus (D_1'D_2)$$

Il faut expliquer clairement la démarche utilisée en inscrivant les théorèmes utilisés à chacune des lignes. Les équations doivent être écrites sur un support électronique (exemple Word). Ceci est devenu une exigence car la grande majorité des photographies sont de piètre qualité, induisant des doutes majeurs dans la correction.

La correction du devoir sera progressive en fonction de l'atteinte de la compétence telle que présentée dans le Tableau 1. Assurez-vous d'être très clair entre chacune des lignes écrites. Identifiez les théorèmes que vous utilisez.

Tableau 1 : Grille d'évaluation du devoir

	Compétence	*GEN420_1*	*GEN420_1*
	Rubrique	*Question 1*	*Question 2*
	Critère	*Connaître et appliquer la démarche d'analyse de problèmes de logique combinatoire menant à la production d'équations logiques réduites.*	*Résoudre algébriquement des équations de logique combinatoire par algèbre de Boole.*
Niveaux	Pondération	5,00	5,00
Excellent	100,00%	Connaît et applique de manière correcte la démarche.	Connaît et applique de manière optimale la démarche de réduction algébrique d'équations de logique combinatoire.
Cible	85,00%	Connaît et applique la démarche avec une erreur mineure.	Connaît la démarche de réduction algébrique d'équations de logique combinatoire, mais l'applique avec une erreur mineure.
Seuil	60,00%	Connaît et applique la démarche avec plusieurs erreurs mineures.	Connaît la démarche de réduction algébrique d'équations de logique combinatoire, mais l'applique avec quelques erreurs mineures.
Non satisfaisant	25,00%	Connaît et applique la démarche avec plusieurs erreurs.	Connaît la démarche de réduction algébrique d'équations de logique combinatoire, mais l'applique avec plusieurs erreurs.
Non initié	0,00%	Ne connaît pas la démarche.	Ne connaît pas la démarche de résolution algébrique d'équations de logique combinatoire par l'algèbre de Boole.

5.2. Validation

Avant de faire ce travail, lire la grille d'évaluation. Le but cette activité est d'expliquer, à haut niveau, votre *démarche* pour le module *Thermo2bin* **seulement**, d'en démontrer sa fonctionnalité par des simulations et de montrer que votre circuit fonctionne sur FPGA. La validation est réalisée **par équipe de 2.** Votre présentation à la validation, prévue à l'horaire (**horarius**), durera un maximum de 3 minutes et comprendra les 4 éléments suivants :

1- *Démarche d'analyse*. La démarche d'analyse doit commencer par expliquer l'objectif global du module *thermo2bin* suivi de votre démarche de conception similaire à ce qui est présenté à l'annexe. Vos explications n'incluent pas de tables de vérité/Karnaugh ni d'équations car elles seront présentées dans le rapport. Un schéma bloc similaire à ce qui est présenté à la figure 1.3 de l'annexe est donc idéal à ce niveau pour accompagner la présentation. Donnez le niveau de détails qui permettra de faire des liens avec ce qu'il faut présenter au point 3. Notez que le code VHDL comme tel sera plutôt évalué dans le rapport. N'y perdez pas de temps à ce moment-ci.

2- *Vérification de Thermo2bin*. Pour cette partie, présentez et expliquez votre liste de tests pour ce module et pourquoi ils confirment la validité de votre circuit en lien avec sa spécification. Il est important d'expliquer comment vous couvrez tous les cas de figure importants sans faire les 2^{12} possibilités de cet encodeur. **Vous devez déposer votre liste de tests sous format PDF avant 17 h le jour précédent la période de validation**.

3- *Exécution, visualisation et interprétation des simulations*. Pour sauver du temps, préparez les tests en VHDL, exécutez la simulation et affichez les chronogrammes avant votre présentation. Faire des changements de fenêtres pour expliquer les chronogrammes. Durant votre présentation, reprenez vos objectifs de tests (point 2) et montrez où ils se trouvent dans le chronogramme simulé. Interprétez la réussite/échec des différents cas de votre séquence de tests.

4- *Exécuter le code sur la carte FPGA*. Les tuteurs vont vérifier votre code fonctionnel sur la carte FPGA.

Tableau 2 : Grille d'évaluation de la validation

	Compétence	*GEN420_1*	*GEN430_1*	*GEN430_1*	*GEN430_1*
	Rubrique	*Démarche d'analyse*	*Vérification*	*Interprétation de chronogrammes*	*Implémentation*
	Critère	*Réaliser une démarche d'analyse nécessaire en vue de réaliser des schémas logiques et/ou rédiger du code VHDL*	*Conçoit et réalise une séquence de tests, identifie correctement les cas d'intérêt et réalise les simulations.*	*Présente et interprète les résultats provenant de chronogrammes.*	*Implémenter et valider un circuit fonctionnel*
Niveaux	Pondération	10,00	10,00	5,00	5,00
Excellent	100,00%	Réalise adéquatement la démarche d'analyse et l'applique avec cohérence en vue de réaliser des schémas logiques et/ou rédiger du code VHDL.	Conçoit et réalise une séquence de tests, identifie correctement les cas d'intérêt et valide le module par simulations.	Présente et interprète parfaitement les résultats provenant de chronogrammes.	Toutes les fonctions logiques sont correctement implémentées. Le circuit est fonctionnel durant la validation.
Cible	85,00%	Réalise adéquatement la démarche d'analyse et l'applique avec quelques problèmes mineurs en vue de réaliser des schémas logiques et/ou rédiger du code VHDL.	Conçoit et réalise une séquence de tests, identifie la majorité des cas d'intérêt et valide le module par des simulations.	Présente et interprète avec une erreur mineure les résultats provenant de chronogrammes.	La majorité des fonctions logiques sont correctement implémentées.
Seuil	60,00%	Réalise les parties essentielles de la démarche d'analyse sans appliquer la démarche complète en vue de réaliser des schémas logiques et/ou rédiger du code VHDL.	Conçoit et réalise partiellement une séquence de tests, identifie partiellement les cas d'intérêt et valide le module par simulations.	Présente et interprète avec plusieurs erreurs mineures les résultats provenant de chronogrammes.	Le circuit fonctionne pour les fonctions essentielles en regard de la spécification.

Non satisfaisant	25,00%	Réalise superficiellement la démarche d'analyse ce qui rend problématique la réalisation des schémas logiques et/ou la rédaction de code VHDL.	Conçoit et réalise une séquence inadéquate de tests, identifie peu de cas d'intérêt et/ou ne valide pas le module par simulations.	Présente et interprète avec des erreurs majeures les résultats provenant de chronogrammes.	Le circuit fonctionne pour quelques fonctions en regard de la spécification.
Non initié	0,00%	Ne réalise pas la démarche.	Ne maîtrise pas le développement d'une séquence de tests, n'identifie pas les cas d'intérêt et ne valide pas le module par simulations.	Ne présente et n'interprète pas les résultats provenant de chronogrammes.	Le circuit ne fonctionne pas

5.3. Rapport d'APP à remettre avant 9h00 (AM) le jour du tutorat de fermeture

Le rapport d'APP contient 2 actions soit la remise du rapport et deuxièmement la soumission du projet Xilinx sur le serveur du département.

5.3.1. Rapport écrit

Le rapport écrit doit être remis **par équipe de 2** et comprend 3 parties :

1- Démarche menant à la réalisation du module thermo2bin.

   a. *Génération des équations et calcul de la fréquence d'opération du module thermo2bin*. Résumez rapidement la démarche globale et vos schémas blocs à haut niveau déjà présentés à votre validation, puis présentez les développements requis pour obtenir les équations simplifiées nécessaires pour la rédaction de votre code en VHDL. Votre solution doit répondre aux spécifications du module *thermo2bin* détaillées dans l'annexe du guide étudiant. Les schémas blocs, table de vérité, tableau de Karnaugh peuvent être mis en annexe. En supposant que toutes les portes logiques de votre schéma ont un délai unique (t_{pHL} et t_{pLH}) de 5 ns, est-ce que votre circuit peut opérer à 20 MHz. Étayez votre explication.

   b. *Implémentation de schémas/code VHDL*. Vous allez ajouter en annexe **seulement** le code VHDL de votre module *Thermo2bin* et de ses sous-modules. Expliquez sommairement la structure du code ; utilisez une figure dans votre texte un peu à la manière de la figure 1.3 de l'annexe. Il faut que tout le code fourni soit autosuffisant pour que nous n'ayons pas à aller le chercher dans votre projet Vivado. Vous pouvez réduire au code nécessaire i.e. enlever les sections en préambule (*entity* et déclaration des signaux) et ne présenter que les sections *architecture*.

2- *Simulation du projet complet*. La démarche démontre les résultats de simulation ayant les entrées et sorties définis à la section 1.3.9 de l'annexe du guide étudiant. Notez qu'il sera très avantageux d'ajouter aussi l'affichage sous forme de caractères tel qu'utilisé dans le laboratoire (disponible dans « */AppCombi_top_tb/uut/inst_aff/o_AFFSSDSim»)*. Il est important d'expliquer comment s'assurer de couvrir tous les cas de figure importants sans faire les 2^{12} possibilités du code thermométrique. Simulez et générez les chronogrammes adéquats. Ajoutez ces chronogrammes dans le rapport tel que spécifié ci-bas. Commentez sur la réussite/échec des différents cas de votre séquence de tests. Utilisez des encadrés numérotés dans les chronogrammes pour faciliter le lien entre le texte et ces derniers.

3- Une démarche d'analyse complète de la compatibilité des signaux entre le FPGA et la DEL D2 complète le rapport écrit.

Vous disposez de **6 pages** pour votre contenu. Ces 6 pages **incluent** le texte, les équations, les chronogrammes et tableaux, mais **excluent** la page titre, la table des matières ainsi que les schémas/code VHDL qui sont en annexe. Le texte doit s'adresser à un professionnel qui connait la matière couverte. Ce professionnel sait comment faire une simplification de tableaux de Karnaugh par exemple. Il faut donc plus s'attarder sur le processus de résolution de problème utilisé et non à expliquer comment fonctionne l'algèbre de Boole ou un tableau de Karnaugh. Le code VHDL peut être en police Consolas 8 et être réduit pour ne conserver que l'essentiel (pas d'entête – voir exemple de code en police Consolas 8 à la

section 1.3.7 et 1.3.8 de l'annexe). Les correcteurs n'ouvriront pas les projets Vivado pour vérifier l'ensemble de votre code VHDL. L'examen pratique fera cette validation. Le rapport doit suivre le guide de rédaction remis en S1. Le rapport d'APP devra être téléversé en un seul fichier **en format PDF** sur le site WEB départemental. Le fichier sera appelé *cip1-cip2.pdf* où *cip* correspond au CIP de chacun des étudiants de l'équipe. Notez la présence du signe « - » entre les deux CIP. Le système n'acceptera pas le dépôt si le CIP de la personne faisant le dépôt n'apparaît pas dans le nom de fichier. La même note sera attribuée aux 2 étudiants. Tout retard sur le dépôt électronique entraînera une pénalité de 20 % par jour.

Le rapport sera corrigé une seule fois et une seule note sera attribuée aux aspects de l'ingénierie, en fonction du Tableau 3 : Grille d'évaluation du rapport écrit. Cependant, la qualité de la communication influence grandement la note attribuée aux aspects de l'ingénierie. Ainsi, des phrases mal construites ou pouvant porter à interprétation, des nombres sans unités ou des graphiques incomplets sont autant d'exemples qui nuisent à la compréhension technique et font perdre des points sur cet aspect. Les correcteurs n'ont pas à interpréter des phrases ou des paragraphes. Soyez donc précis et concis. La qualité de la communication sera évaluée selon le Tableau 4 (Évaluation de la communication). 15% pourront être retranchés à la note du rapport en fonction de la grille de correction. L'atteinte du niveau *cible* dans la grille assure de ne pas perdre de points, mais l'atteinte d'un niveau inférieur au niveau *cible* fait automatiquement perdre des points.

L'évaluation du rapport d'APP contribue à l'évaluation des éléments de compétence de l'unité. Il s'agit donc d'une évaluation sommative, c'est-à-dire que le résultat de l'évaluation sera consigné au dossier scolaire de l'étudiant et utilisé dans le calcul de sa note finale. Toutefois, pour permettre à l'étudiant d'apprendre de ses erreurs, son rapport corrigé lui sera remis, avec une grille lui permettant d'apprécier son niveau de compétence.

Il est important de noter qu'il est de la responsabilité de l'étudiant signataire du rapport de s'assurer de l'exactitude de la valeur de chaque élément de solution et de la qualité et de l'uniformité de l'ensemble du contenu de son rapport. N'oubliez pas que, lors de l'évaluation sommative (examen à la fin de l'unité), vous allez être évalués **de façon individuelle** sur les compétences mises en œuvre pour l'élaboration de ce rapport. Vous êtes donc réputé pouvoir résoudre l'ensemble de la problématique de façon individuelle, de même que tout problème relié aux connaissances nouvelles à acquérir durant cette unité.

Même si les travaux sont remis en équipe de 2 personnes, vous devez être en mesure de résoudre la problématique de façon individuelle pour être en mesure de réussir les examens.

5.3.2. Téléversement du projet Xilinx

Votre rapport doit être accompagné du téléversement de votre projet Xilinx sur le site WEB de l'APP :

- Renommer le fichier *.bit* d'implantation de votre solution par le nom *cip1.bit* afin de le préserver avant de faire d'autres commandes dans les processus du projet Vivado qui pourraient l'effacer.
- Pour limiter la dimension du projet archivé, vérifiez la dimension des fichiers *.wdb* créés pour les simulations qui peuvent devenir très volumineux. Éventuellement faites une réinitialisation de la simulation avant d'archiver pour réduire la dimension.
- Générer une archive .zip (dans Vivado : menu file/project/Archive) ou comprimer le répertoire du projet en suivant une démarche équivalente. Le fichier téléversé est un fichier « zippé » comprenant tous les fichiers sources nécessaires à l'implémentation et la simulation et le fichier *.bit*.

Le fichier zippé doit comporter le nom suivant *cip1-cip2.zip* où *cipX* correspondent à vos *cips*.

Tableau 3 : Grille d'évaluation du rapport écrit

	Compétence	*GEN420_1*	*GEN420_1*	*GEN430_1*	*GEN430_1*	*GEN430_1*	*GEN430_1*
	Rubrique	*Démarche*	*Réalisation*	*Vérification*	*Interprétation de chronogrammes*	*Technologies*	*Technologies*
	Critère	*Réaliser une démarche d'analyse nécessaire en vue de dessiner des schémas logiques et/ou rédiger du code VHDL*	*Réaliser des schémas logiques et/ou code VHDL*	*Conçoit et réalise une séquence de tests unitaires, identifie correctement les cas d'intérêt et réalise les simulations.*	*Présente et interprète les résultats provenant de chronogrammes.*	*Résoudre des problèmes de délais de propagation*	*Résoudre des problèmes de compatibilité de technologies et de fanout*
Niveaux	Pondération	10,00	15,00	10,00	5,00	5,00	5,00
Excellent	100,00%	Réalise adéquatement la démarche d'analyse et l'applique avec cohérence en vue de dessiner des schémas logiques et/ou rédiger du code VHDL.	Réalise parfaitement des schémas logiques et/ou code VHDL.	Conçoit et réalise une séquence de tests, identifie correctement les cas d'intérêt et valide le module par simulation.	Présente et interprète correctement les résultats provenant de chronogrammes.	Résout adéquatement des problèmes de calcul de délais de propagation en émettant des hypothèses adéquates.	Résout adéquatement des problèmes de compatibilité de technologies et *fanout* en émettant des hypothèses adéquates.
Cible	85,00%	Réalise adéquatement la démarche d'analyse et l'applique avec quelques problèmes mineurs en vue de dessiner des schémas logiques et/ou rédiger du code VHDL.	Réalise des schémas logiques et/ou code VHDL avec une erreur mineure.	Conçoit et réalise une séquence de tests, identifie la majorité des cas d'intérêt et valide le module par simulation.	Présente et interprète avec une erreur mineure les résultats provenant de chronogrammes.	Résout des problèmes de calcul de délais de propagation, mais n'utilise pas les bonnes données.	Résout des problèmes de compatibilité de technologies et *fanout* en émettant des hypothèses adéquates, mais n'utilise pas les bonnes données.
Seuil	60,00%	Réalise les parties essentielles de la démarche d'analyse sans appliquer la démarche complète en vue de dessiner des schémas logiques et/ou rédiger du code VHDL.	Réalise des schémas logiques et/ou code VHDL avec des erreurs mineures.	Conçoit et réalise partiellement une séquence de tests, identifie partiellement les cas d'intérêt et valide le module par simulation.	Présente et interprète avec plusieurs erreurs mineures les résultats provenant de chronogrammes.	Résout partiellement des problèmes de calcul de délais de propagation et émet des hypothèses partielles.	Résout partiellement des problèmes de compatibilité de technologies et *fanout* et émet des hypothèses partielles.
Non satisfaisant	25,00%	Réalise superficiellement la démarche d'analyse ce qui rend problématique le dessin des schémas logiques et/ou la rédaction du code VHDL.	Réalise des schémas logiques et/ou code VHDL avec des erreurs majeures.	Conçoit et réalise une séquence inadéquate de tests, identifie peu de cas d'intérêt et/ou ne valide pas le module par simulation.	Présente et interprète avec des erreurs majeures les résultats provenant de chronogrammes.	Résout partiellement des problèmes de délais de propagation et émet des hypothèses erronées.	Résout partiellement des problèmes de compatibilité de technologies et *fanout* et émet des hypothèses erronées.
Non initié	0,00%	Ne connait pas la démarche.	Ne maîtrise pas la réalisation de schémas logiques et/ou code VHDL.	Ne maîtrise pas le développement d'une séquence de tests, n'identifie pas les cas d'intérêt et ne valide pas le module par simulation.	Ne présente et n'interprète pas les résultats provenant de chronogrammes.	Ne résout pas des problèmes de délais de propagation.	Ne résout pas des problèmes de compatibilité de technologies et de *fanout.*

Tableau 4 : Évaluation de la communication écrite

Rubrique	*Organisation*	*Présentation*	*Langue qualité*
Critère	*Organiser et présenter de l'information pertinente*	*Présenter des communications graphiques et/ou un support visuel de qualité*	*Communiquer dans une langue de qualité, à l'écrit comme à l'oral*
Excellent	Organise efficacement l'information pour en faciliter la compréhension. L'information est pertinente et complète au regard des objectifs de la communication. Le sujet est bien délimité et présenté avec concision	Présente des communications graphiques qui appuient le texte, sont faites selon l'état de l'art et sont accompagnées d'un titre évocateur. Recourt à un support visuel qui appuie efficacement ses propos et veille à ce que son support soit sobre, visible, équilibré et d'allure professionnelle	Fait des phrases complètes et bien structurées. Respecte les règles d'orthographe, de syntaxe et de grammaire élémentaires, en plus d'utiliser une terminologie et un vocabulaire tout à fait appropriés
Cible	Organise adéquatement l'information pour en permettre la compréhension. L'information est pertinente et complète au regard des objectifs de la communication	Présente des communications graphiques qui appuient le texte et qui sont généralement faites selon l'état de l'art. Recourt à un support visuel qui appuie bien ses propos et veille à ce que son support soit sobre, visible et équilibré	Fait généralement des phrases complètes et bien structurées. Respecte les règles d'orthographe, de syntaxe et de grammaire élémentaires, en plus d'utiliser une terminologie et un vocabulaire généralement appropriés
Seuil	Organise l'information pour en permettre la compréhension. L'information est pertinente au regard des objectifs de la communication, mais des éléments importants sont mal présentés	Présente des communications graphiques qui appuient le texte, mais celles-ci ne sont pas toujours faites selon l'état de l'art. Recourt à un support visuel qui appuie minimalement ses propos	Fait encore des erreurs d'orthographe, de syntaxe et de grammaire, mais celles-ci ne nuisent pas à la compréhension de son texte ou discours. Utilise une terminologie et un vocabulaire minimalement appropriés
Non satisfaisant	Organise minimalement l'information pour en permettre la compréhension. L'information est pertinente au regard des objectifs de la communication, mais quelques éléments importants sont absents	Présente des communications graphiques qui appuient minimalement le texte. Recourt à un support visuel dont l'allure devrait être améliorée	Fait généralement des phrases complètes et bien structurées, mais celles-ci comportent des erreurs d'orthographe, de syntaxe ou de grammaire qui nuisent parfois à la compréhension de son texte ou discours.
Non initié	N'est pas en mesure d'organiser l'information pour en permettre la compréhension. Éprouve de la difficulté à sélectionner l'information pertinente au regard des objectifs de la communication	Ne présente pas des communications graphiques qui appuient le texte. Recourt à un support visuel qui n'est pas adapté	Éprouve de la difficulté à faire des phrases complètes et bien structurées, de même qu'à respecter les règles grammaticales élémentaires, ce qui nuit à la compréhension de son texte ou discours. N'utilise pas une terminologie et un vocabulaire appropriés

5.4. (Optionnel) Schéma pour remettre à la fin du tutorat de fermeture

Schéma de concept sur les circuits logiques combinatoires qui doivent répondre à la question suivante :

- Quels concepts devez-vous utiliser pour résoudre une problématique en logique combinatoire à partir des spécifications de conception et arriver à une réalisation physique fonctionnelle ?
  <u>Note</u> : Inscrire votre nom, votre numéro de *cip* sur chaque schéma remis.

5.5. Distribution des points en fonction des activités pédagogiques

Activité	*Élément de compétence*	*Devoir*	*Validation au laboratoire*	*Rapport d'APP*	*Examen Sommatif*	*Examen pratique*	***Totaux***
GEN420_1	Modéliser et résoudre un problème de logique combinatoire et séquentielle à l'aide de représentations mathématiques de l'information discrète et par la synthèse des équations booléennes.	45			135	-	**180**
		10	10	25			
GEN430_1	Concevoir et réaliser des systèmes numériques combinatoires à partir de spécifications.	45			55	80	**180**
		-	20	25			

5.6. Modalité des examens

L'APP comporte un examen pratique sur Vivado, où vous devez apporter votre carte FPGA, et un examen théorique. Ce dernier comporte une partie Moodle et une partie papier. Les calculatrices ne sont pas permises à l'examen. Vous devez être en mesure de faire les transformations de base par des calculs manuels. Il est attendu que la personne étudiante connaisse ses tables de multiplication.

6. Formation à la pratique procédurale #1

6.1. Buts de l'activité

On veut ici mettre en pratique les procédures requises pour :

- Représenter et modéliser de l'information discrète ;
- Convertir un nombre d'une base à une autre ;
- Effectuer des additions et des soustractions en binaire et en complément à 2 ;
- Représenter une expression logique algébriquement ou par table de vérité ;
- Écrire des expressions logiques sous la forme somme de produits ou produit de sommes ;
- Simplifier des expressions logiques algébriquement ou avec un tableau de Karnaugh ;
- Établir le complément et le dual d'une expression logique.

Consultez le guide de lecture pour localiser rapidement les lectures à réaliser.

À faire avant le procédural (solutions sur le site web) :

1. Représentez ces nombres sans utiliser la calculatrice :
   a. $1000\ 1011_2 = ?_{10} = ?_8 = ?_{16} = ?_{BCD}$
   b. $125_{10} = ?_{\text{Amplitude signée}} = ?_{\text{ Complément 1}} = ?_{\text{Complément 2}}$
   c. $-12.55_{10} = ?_{\text{IEEE single precision Floating point representation}}$ *
2. Simplifiez l'expression : $F = A \cdot B + A \cdot B \cdot C' \cdot D + A \cdot B \cdot D \cdot E' + A' \cdot B \cdot C' \cdot E + A' \cdot B' \cdot C' \cdot E$

6.2. Problèmes à résoudre

La résolution des problèmes suivants va vous permettre de mettre en pratique les connaissances acquises durant votre étude et développer vos compétences en logique combinatoire.

1. (GEN 420) Indiquez s'il y a débordement ou non des nombres suivants représentés en complément à 2 sur 8 bits (5 minutes) :
   a. 11010100 + 10101011
   b. 01011101 + 00100001

2. (GEN 420) Démontrez les égalités (20 minutes) :
   a. (M'+R'(NO)')'+(Q+P+N)'+(P'+R'+M')'+Q'OMP' = N M O + N'Q'P'+ M R
   b. (AB'+(ABC)')(A+(BC)') ⊕ A = ABC + A'B' + A'C'

* Note, la notion de IEEE single precision n'est pas à l'examen mais elle doit être vue pour des sessions futures. Il faut regarder comment cette notation est créée.

3. (GEN 420) Soit la fonction

$$F= \Pi_{w,x,y,z} (1,4,7)$$

   a. Écrire la somme et le produit canonique de cette équation et simplifier à l'aide des tableaux de Karnaugh (15 minutes)

4. (GEN 420) Un monte-charge doit permettre le levage de masses comprises entre 10 kg et 60 kg. Il comporte une plateforme reposant sur des ressorts. Suivant l'importance de la charge à soulever, trois contacts réglables peuvent être actionnés (25 minutes) :

   a. À vide, aucun des trois contacts n'est actionné. Le monte-charge doit fonctionner.

   b. Pour des charges comprises entre 5 et 10 kg, seul le contact « a » est actionné. Le monte-charge ne doit pas fonctionner.

   c. Pour des charges comprises entre 10 et 60 kg, les contacts « a » et « b » sont actionnés. Le monte-charge doit fonctionner.

   d. Pour des charges supérieures à 60 kg, les trois contacts « a », « b » et « c » sont actionnés. Le monte-charge ne doit pas fonctionner.

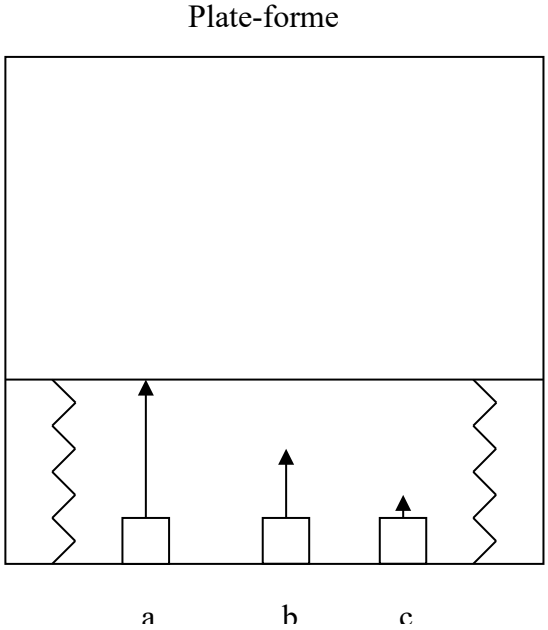


Figure 3. Schéma du monte-charge et ses capteurs

1. Réaliser le schéma logique du système permettant de savoir si le monte-charge peut ou non fonctionner.
2. Écrire la fonction en VHDL sous forme d'équation (ex. A XOR B)
3. On vous demande de vérifier la fonctionnalité du circuit par une simulation. Quels cas allez-vous tester ?

5. (GEN 420) Écrire les équations pour un additionneur 1 bit (préparation au pré-laboratoire)
    a. Faire une description fonctionnelle haut niveau du module (entrées, sorties, fonction)
    b. Écrire les équations des fonctions S_i et C_o sous forme de somme de produits puis transformer les expressions pour faire apparaître des opérateurs « OU exclusif ou XOR »
6. (GEN 420) À partir de la Table I fournie à la page 14 de votre guide étudiant, simplifiez le segment *C* en utilisant un tableau de Karnaugh (SOP) (25 minutes). Écrire la fonction en VHDL en utilisant la fonction case *et* With-Select.

7. Formation à la pratique en laboratoire #1

7.1. Buts de l'activité

Mettre en pratique expérimentalement les procédures que vous avez développées jusqu'à présent. Cette activité fait suite à l'atelier de formation sur l'environnement de développement Xilinx que vous avez déjà suivi. Nous vous proposons maintenant de conduire un projet de logique combinatoire (le sujet du laboratoire) de la création du projet jusqu'à la phase de simulation fonctionnelle suivie de l'implémentation sur la carte Xilinx. Comme vous allez le constater, la résolution de ce laboratoire constitue une étape importante de la solution de la problématique de cette unité.

Vous n'avez pas de rapport de laboratoire à écrire.

7.2. Description du laboratoire : Module additionneur 4 bits

Vous devez réaliser un module capable d'additionner 2 mots de 4 bits. Vous utiliserez pour cela l'algorithme classique de propagation de la retenue et la numération positionnelle.

Pour mener à bien et rapidement ce travail, nous vous suggérons de le découper en deux étapes distinctes : un prélab et une expérimentation.

Pré-Lab : Cette étape se réalise complètement sur papier (cahier de laboratoire) et avant de se présenter pour la seconde étape. Elle est la suite immédiate du numéro préparatoire du procédural 1.

1- Faire une description fonctionnelle haut niveau du module (entrées, sorties, fonction). Utiliser la notation suivante :
   - Entrées : mots *X* et *Y*, $X = (x_3, x_2, x_1, x_0)$ et $Y = (y_3, y_2, y_1, y_0)$ et retenue d'entrée *Ci*
   - Sorties : somme $S = (s_3, s_2, s_1, s_0)$ et retenue C_O.

2- Montrer que l'algorithme d'addition en numération positionnelle peut se réaliser avec une succession d'additionneurs élémentaires pour 1 bit et identiques opérant chacun sur une position x_i, y_i, c_i des mots d'entrée ;

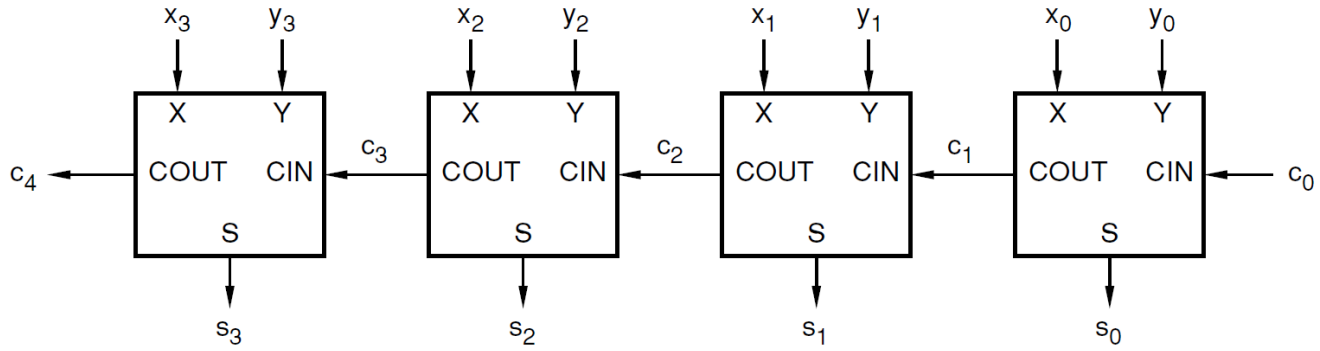


Figure 4. Additionneur en numération positionnelle

3- Reprendre le numéro préparatoire du procédural 1 et décrire un additionneur élémentaire complet 1 bit, pour cela :
   - Écrire les équations des fonctions S_i et C_o sous forme de somme de produits puis transformer les expressions pour faire apparaître des opérateurs « OU exclusif ou XOR ».

- Dessiner le schéma logique des fonctions S_i et C_o en employant des portes XOR, AND et OR.
- Écrire les équations logiques correspondantes sous forme VHDL.

4- Dessiner sur papier à partir du module additionneur 1 bit le schéma complet d'un additionneur 4 bits.

Expérimentation : Cette étape demande d'utiliser le logiciel Xilinx Vivado ;

1- Utiliser le projet de la problématique de l'APP (voir sur le site WEB) auquel vous ajouterez 3 nouveaux modules appelés Add1bitA, Add1bitB et Add4bits.

2- Les deux modules à 1 bit ont les mêmes entrées X, Y, C_i et sorties S, C_o.

3- Le module Add4bits possède les entrées X(3 :0), Y(3 :0) et C_i, et en sortie S(3 :0) et C_o. L'objectif ici est de vous pratiquer à ajouter des modules à un circuit existant.

4- Réaliser la version de l'additionneur 1 bit ci-dessus dans les modules correspondants ;
   a. Add1bitA : utiliser une architecture comportementale basée sur une table de vérité avec une fonction VHDL CASE.
   b. Add1bitB : utiliser une architecture comportementale basée sur des équations booléennes ex. « y <= A AND B ; ».

5- Valider votre solution en effectuant une simulation fonctionnelle du module additionneur 1 bit. Partez du banc de test générique de l'atelier (fourni sur la page web de l'APP) et modifiez-le pour générer un stimulus similaire ou identique à celui suggéré à la figure suivante :

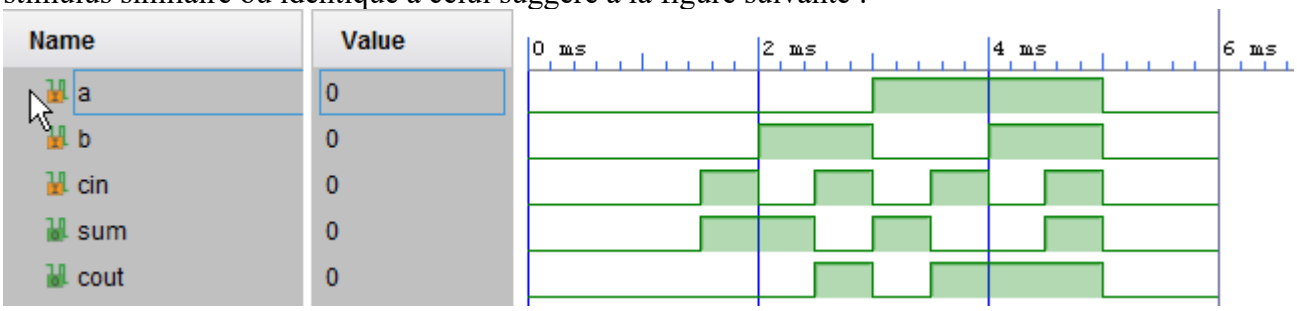


Figure 5. Simulation d'un additionneur à 4 bits

6- Add4bits : Utilisez 2 modules Add1bitA et 2 modules Add1bitB pour réaliser un additionneur 4 bits par une approche hiérarchique modulaire avec propagation de la retenue à l'aide d'instanciations en VHDL (Port Map). Voir exemple de l'atelier Xilinx pour instancier des modules ;

7- Valider votre solution de l'Add4bits en effectuant une simulation fonctionnelle du module additionneur 4 bits;

Réponse : Une simulation du module Add4bits dans le cas d'une addition de 0 + 15, 12 + 2, 13 + 15 et 0 + 0 est présenté à la figure 8.2. Une première série sans retenue à l'entrée et une seconde série identique avec retenue à l'entrée. Les opérandes et la somme sont présentés en nombre de 4 bits en format hexadécimal pour chacun des cas de tests individuels. Les valeurs sont maintenues pour 500 µs afin de présenter un chronogramme temporel lisible.

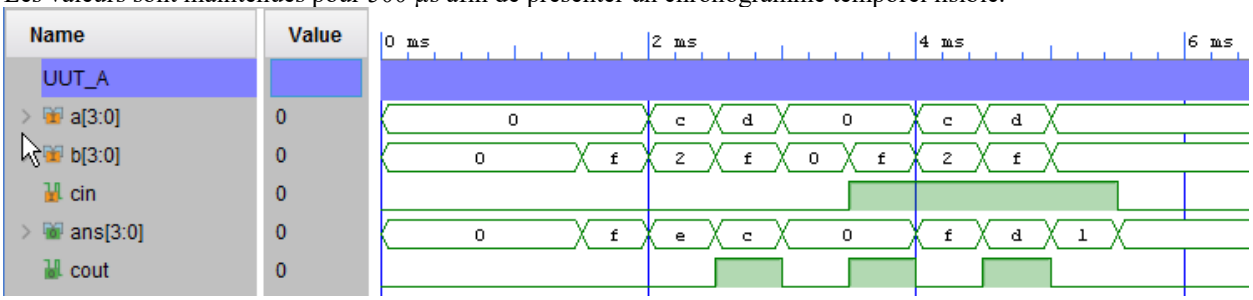


Figure 6. Simulation d'un additionneur à 4 bits

7.3. Intégration initiale du module *Add4bits* à la problématique.

Réalisez le montage de la figure suivante :

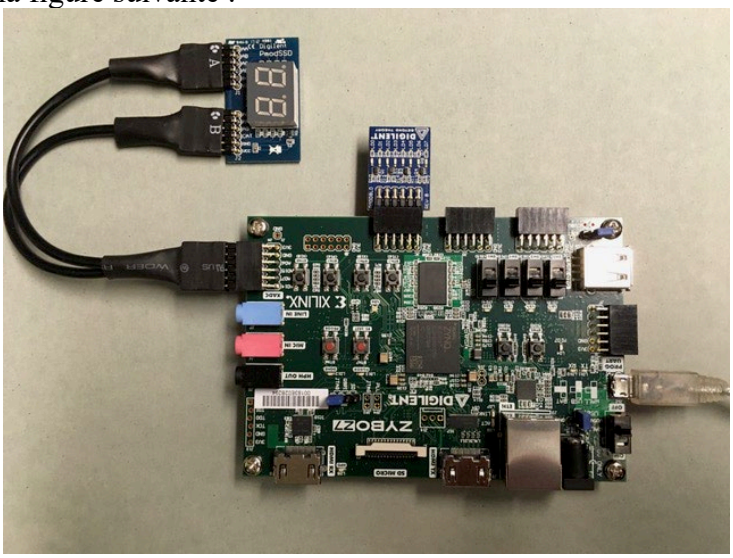


Figure 7. Montage pour additionneur 4 bits

Porter attention au branchement des connecteurs dans la disposition illustrée. Assurez-vous de la polarité des connecteurs pour avoir le retour de courant au bon endroit.

Programmez votre carte Zybo-Z7 en réalisant les étapes suivantes :

1- Ajoutez votre fichier *Add4bits* dans le projet Vivado du laboratoire, si ce n'est pas déjà fait. Validez que le circuit *AppCombi_top* est identifié comme *« top module »* de la hiérarchie *Design Sources* du projet *Vivado*.

2- Dans le code VHDL, ajoutez la déclaration de votre le « component » *Add4bits* dans *AppCombi_top* et instanciez-y une unité de ce module avec les branchements (*port map*) sur les signaux *d_opa, d_opb*, *d_cin*, *d_sum, d_cout* déjà préparés dans le circuit. Les interrupteurs SW(3 :0) et BTN (3 :0) seront utilisés comme entrées et la somme sera branchée sur iAFF0(3 :0) de l'instance inst_aff comme le démontre la Figure 7 ci-dessous. Cout pourra être branché sur o_led6_r et sur le LSB de i_AFF1(3 :0).

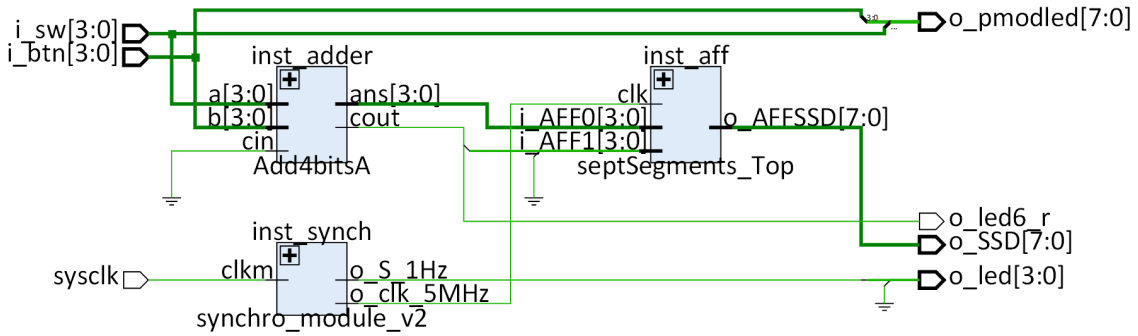


Figure 8. Schéma équivalent au module top après l'insertion de votre additionneur.

3- Assurez-vous de la présence du fichier de contraintes *AppCombi_top.xdc* qui spécifie la position physique de vos signaux sur les pattes du FPGA sur le PCB (signaux SW, BTN et les modules Pmod LED et SSD).
4- Compilez votre projet (*generate bitstream*) et programmez la carte FPGA.
5- Validez votre additionneur à l'aide des interrupteurs et boutons. Pour cette partie, vous devez tenir compte que votre addition est en complément à 2 mais affichée en hexadécimal.
6- (possible de faire à la maison; compétence requise pour le rapport) Simulez et validez votre nouveau circuit complet avec le banc de test fourni dans le projet. Avec ce banc de test, vous pouvez ajouter l'afficheur 7-segments sous forme de chaine de caractères au chronogramme†. Le signal spécial, une sortie non connectée du module d'affichage instancié dans le *Top* de logique, est visible dans la hiérarchie à « */AppCombi_top_tb/uut/inst_aff/o_AFFSSDSim* ». Ce signal change au même rythme que votre logique combinatoire et donne une photo de ce qu'on verrait sur l'afficheur physique. Dans les faits, o_AFFSSD[7 :0] contrôle les deux chiffres de l'afficheur physique en basculant entre les deux aux 500 µs. On verra donc sur ce signal seulement un chiffre sur deux, et avec un changement très lent. Fiez-vous donc sur le signal de substitution pour éviter de longs temps de simulation. Voici un exemple de ce que devrait ressembler la simulation.

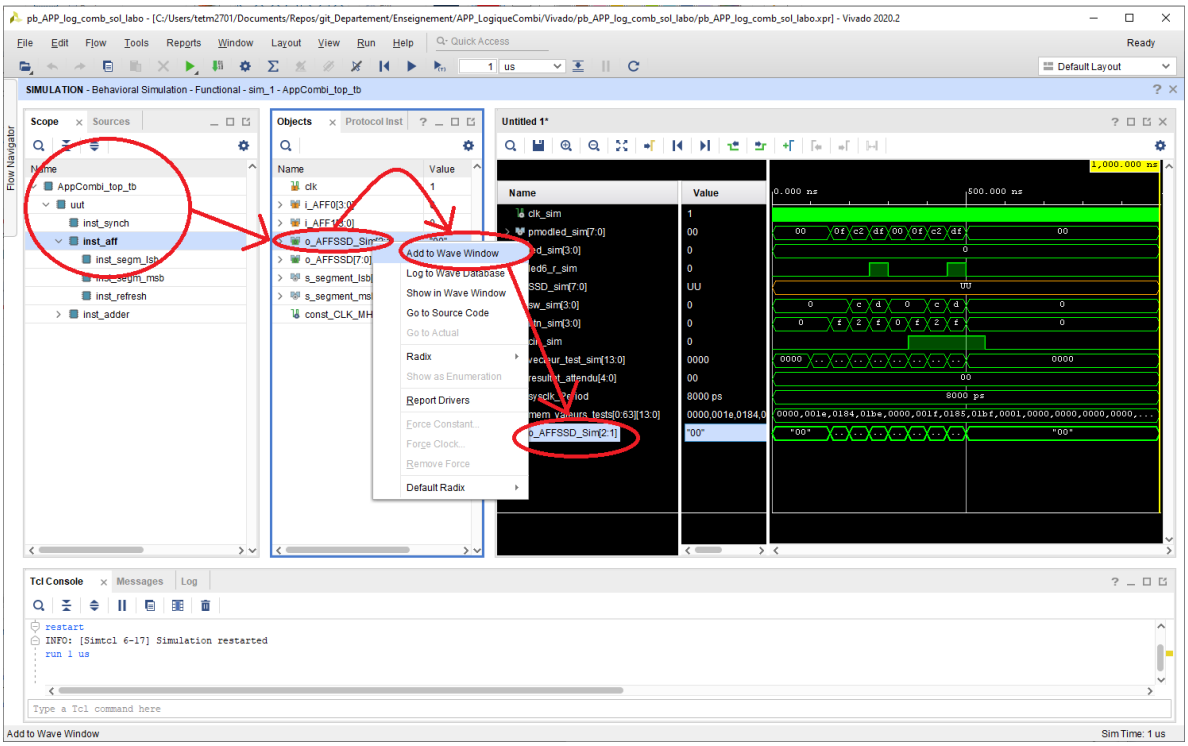


Figure 9. Exemple d'ajout de signaux pour la simulation dans une hiérarchie

† Pour ceci, revenir à la page 53 de l'atelier Vivado où l'ajoute des signaux à visualiser.

8. Formation à la pratique procédurale #2

8.1. Buts de l'activité

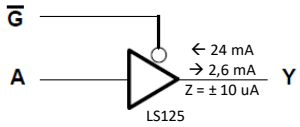
Dans cette activité, on veut mettre en pratique les procédures requises pour :

- Représenter l'information binaire au niveau physique ;
- Tenir compte des caractéristiques électriques et temporelles des composants logiques dans la conception d'un circuit combinatoire ;
- Interpréter les fiches techniques d'une famille logique ;
- Dessiner un schéma logique selon les standards ;
- Calculer les temps de propagation dans un circuit logique combinatoire ;
- Utiliser des fonctions standards dans la solution d'un problème de logique combinatoire ;
- Mettre en œuvre le concept de bus ;
- Analyser le comportement transitoire d'un circuit combinatoire.

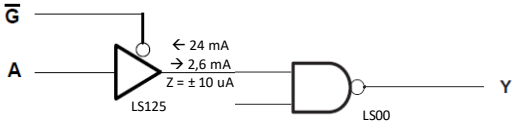
8.2. Problèmes à résoudre

La résolution des problèmes suivants vous permettra de mettre en pratique les connaissances acquises durant votre étude et développer vos compétences en logique combinatoire.

1. (GEN 430) Est-ce qu'on peut alimenter 20 entrées de FPGA à partir d'un circuit SN74LVC1G04DBVR lorsque les circuits sont alimentés à 3,3 Volts ? (voir la spécification du manufacturier sur la page du site WEB) (15 minutes)
2. (GEN 430) Un 74LS125 est un buffer à trois états. Lorsqu'il est activé, la sortie peut tirer (sink) 24 mA dans un état bas et fournir (source) 2,6 mA dans un état élevé. Lorsqu'il est désactivé, la sortie présente un courant de fuite de ± 20 µA (le signe représentant le niveau de tension de sortie).



Supposez qu'on veuille multiplexer plusieurs paires de 74LS125 / 74LS00 branchés comme suit



sur un seul bus selon la figure suivante :

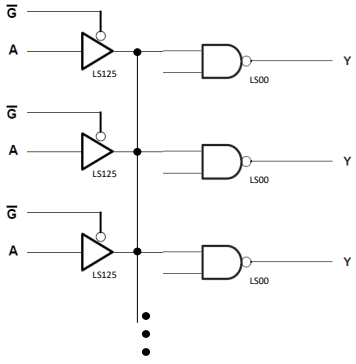


Figure 10. Schéma logique problème LS125 – LS00

Ceci correspond à un bus de 1 bit. Quel serait le nombre maximum de paires 74LS125/74LS00 que l'on pourrait brancher sur ce bus sans excéder les spécifications du manufacturier ? (25 minutes) ?

Note, la convention du sens des courants défini un courant négatif s'il sort d'un dispositif et positif s'il entre.

3. (GEN 430) À partir du circuit suivant,

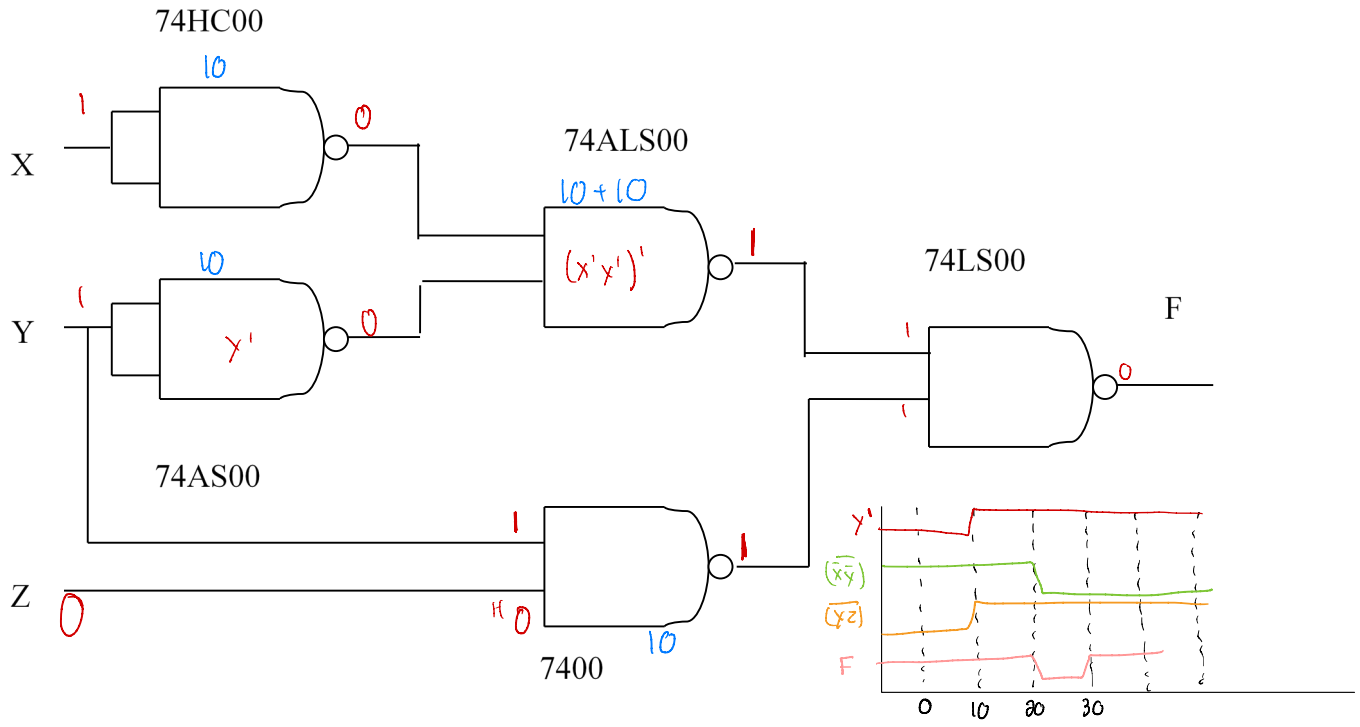


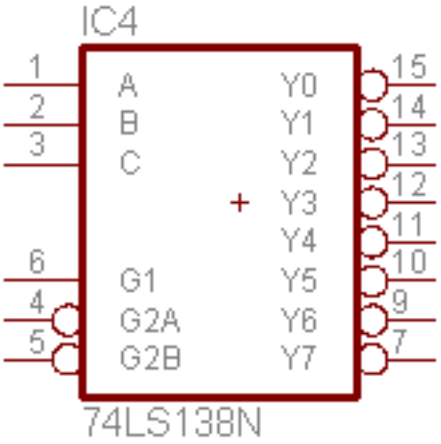
Figure 11. Schéma logique avec portes de différentes technologies

a. Déterminez la fonction F
b. Déterminez le délai de propagation maximal en supposant les caractéristiques suivantes :
   HC00 t_{pHL} = 18 ns, t_{pLH} = 18 ns
   AS00 t_{pHL} = 4 ns, t_{pLH} = 4.5 ns
   ALS00 t_{pHL} = 8 ns, t_{pLH} = 11 ns
   00 t_{pHL} = 15 ns, t_{pLH} = 22 ns
   LS00 t_{pHL} = 15 ns, t_{pLH} = 15 ns
c. En supposant que les niveaux logiques sont adéquats, que se passe-t-il lorsque Y passe de 1 à 0 alors que X et Z demeurent respectivement à 0 et 1 ? Pour cette question, simplifiez les délais de propagation à un délai simple $t_{pHL} = t_{pLH}$ = 10 ns. Comment corriger cet effet ? (25 minutes) :

4. (GEN 430) Le 74LS138 (schéma et circuit à la page suivante) est un circuit intégré d'intégration moyenne effectuant un décodage 3 à 8. Dessinez un diagramme démontrant comment ce circuit peut être utilisé pour créer un décodeur 4 à 16. Identifiez les signaux d'entrée, de sortie et de contrôle. (20 minutes)

5. (GEN 430) Créer un multiplexeur 8 à 1 à l'aide d'un 74LS138 et de buffers 3 états comme le LS125 du numéro 2 de ce procédural. Expliquez votre démarche.

Voici le schéma et la table de vérité du circuit décodeur 74LS138.



Function Tables

DM74LS138

Inputs					Outputs							
Enable		Select										
G1	G2 (Note 1)	C	B	A	YO	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

DM74LS139

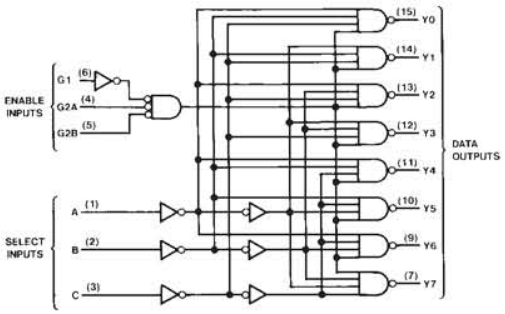
Inputs			Outputs			
Enable	Select					
G	B	A	Y0	Y1	Y2	Y3
H	X	X	H	H	H	H
L	L	L	L	H	H	H
L	L	H	H	L	H	H
L	H	L	H	H	L	H
L	H	H	H	H	H	L

H = HIGH Level
L = LOW Level
X = Don't Care

Note 1: G2 = G2A + G2B

Logic Diagrams

DM74LS138



DM74LS139