

Exercices TD 2 – *Simplification de fonctions logiques et circuits combinatoires*

• **Exercice 1** – [1]* Propriétés des portes logiques :

1.1 Sous quelles conditions une porte OR permettra le passage d'un signal logique vers la sortie sans modification ?

1.2 Répétez l'item précédant pour une porte AND.

1.3 Répétez pour une porte NAND.

1.4 Répétez pour une porte NOR.

1.5 Concevez un circuit logique qui fait qu'un signal A en entrée puisse passer à la sortie uniquement lorsque l'entrée de commande B est à un niveau BAS tandis que l'entrée de commande C est à un niveau HAUT ; sinon, la sortie du circuit sera à un niveau BAS.

• **Exercice 2** – Dans un immeuble de trois étages, on souhaite concevoir un circuit logique pour contrôler l'ouverture de la porte d'un ascenseur. Les variables d'entrée sont E0, E1, E2 et M, où :

- M indique que l'ascenseur est en mouvement lorsqu'elle vaut 1.
- E0, E1, E2 indiquent que l'ascenseur est positionné aux étages 0, 1 et 2 lorsqu'elles valent 1, respectivement.

Vous devez concevoir un circuit combinatoire avec sortie P qui indique – lorsqu'elle est en état haut – que l'ascenseur doit ouvrir la porte. L'ascenseur ouvre sa porte s'il est positionné dans un étage et quand il n'est pas en mouvement. Veuillez aussi noter que l'ascenseur est présent seulement dans un seul étage à la fois.

2.1 Obtenir la table de vérité de ce problème.

2.2 Écrire les formes canoniques SOP et PDS.

2.3 Simplifier ces fonctions avec la table de Karnaugh pour la forme SOP et PDS.

2.4 Concevez deux circuits (logigrammes) correspondants avec portes logiques OR (OU), AND (ET) et NOT (NON).

• **Exercice 3** – Nous voulons concevoir un circuit combinatoire qui calcule le reste de la division par 5 d'un nombre compris entre $(0)_{10}$ et $(9)_{10}$. Ce circuit reçoit en entrée le numéro pour la division en "Binary Coded Decimal" (BCD) avec 4 bits et le résultat est aussi donné en BCD avec 4 bits.

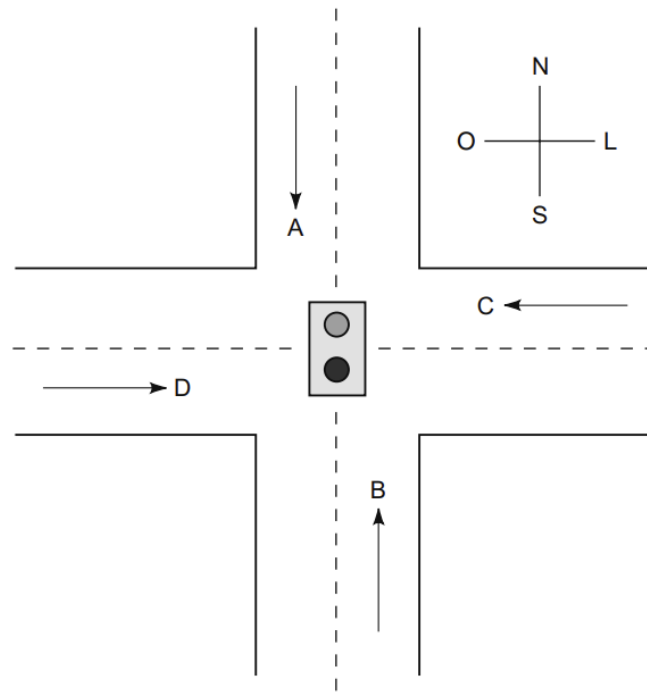
3.1 Calculez la table de vérité pour ce problème.

3.2 Obtenez les formes canoniques SOP et PDS pour chacune des sorties.

3.3 Effectuez la simplification de la fonction logique à l'aide de cartes de Karnaugh pour chacune des sorties.

3.4 Dessinez le logigramme pour ce circuit.

- **Exercice 4** – [1]* Nous voulons proposer un circuit numérique pour contrôler le trafic routier d'un croisement (l'intersection d'une autoroute avec une voie d'accès) avec deux feux à deux couleurs (vert ou rouge) comme dans la figure qui suit:



Des capteurs détecteurs de véhicules sont placés le long des voies C et D (sur l'autoroute) et sur les voies A et B (voie d'accès). Les sorties de ces capteurs seront au niveau BAS (0) lorsqu'aucun véhicule n'est présent et au niveau HAUT (1) lorsqu'un véhicule est présent. Les deux feux de circulation à l'intersection sont commandés selon la logique suivante :

- Le feu dans la direction est-ouest (L-O) sera vert lorsque les deux voies C et D seront occupées.
- Le feu dans la direction est-ouest sera vert chaque fois que les voies C ou D seront occupées, mais avec A et B libres.
- Le feu dans la direction nord-sud (N-S) sera vert chaque fois que les deux voies A et B seront occupées, mais que C et D seront libres.
- Le feu dans la direction nord-sud sera aussi vert lorsque les voies A ou B seront occupées tandis que les deux voies C et D seront vides.
- Le feu dans la direction est-ouest sera vert lorsqu'il n'y aura aucun véhicule présent.

En utilisant les sorties des capteurs A, B, C et D comme entrées, vous voulez concevoir un circuit logique combinatoire pour contrôler les deux feux. Chaque feu correspond à une sortie N-S (F1) et L-O (F2), qui seront au niveau HAUT lorsque la lumière correspondante est verte.

- 4.1 Calculez la table de vérité pour ce problème.
- 4.2 Obtenez les formes canoniques SOP et PDS pour chacune des sorties des feux F1 et F2.
- 4.3 Effectuez la simplification des fonctions logiques pour chacune des sorties.
- 4.4 Dessinez le logigramme pour ce circuit.

([1]*) Exercices adaptés de “Circuits numériques : théorie et applications” / Ronald J. Tocci.