CEG 3555: Systèmes Numériques II
(Automne 2023)
Prof. Rami Abielmona
Solutions Possibles pour Devoir #2: *Circuits Arithmétiques*

3 Octobre, 2023

Question I

Cette question se concerne avec la construction d'un additionneur ternaire demi- et complet. Les parties a et b sont questions 3.25 et 3.26 de votre manuel.

Partie a

L'additionneur-demi ternaire peut être définé en utilisant des signaux encodés en binaire comme suit:

A		*B*		*Report*	*Somme*	
a_1	a_0	b_1	b_0	c_{out}	s_1	s_0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
1	0	0	0	0	1	0
1	0	0	1	1	0	0
1	0	1	0	1	0	1

Les 7 combinaisons (sur 16) restantes où $a_1 = a0 = 1$, ou $b_1 = b_0 = 1$, peuvent êre traitè comme des conditions *don't care*. Alors, les expressions de coût minimum sont:

$$c_{out} = a_0 \cdot b_0 + a_1 \cdot b_1 + a_1 \cdot b_0 \quad (1)$$

$$s_1 = a_0 \cdot b_0 + \overline{a_1} \cdot \overline{a_0} \cdot b_1 + a_1 \cdot \overline{b_1} \cdot \overline{b_0} \quad (2)$$

$$s_0 = a_1 \cdot b_1 + \overline{a_1} \cdot \overline{a_0} \cdot b_0 + a_0 \cdot \overline{b_1} \cdot \overline{b_0} \quad (3)$$

Partie b

L'additionneur-complet ternaire peut être défiué par la table de vérité démontrée dans table 1.

c_{in}	A	B	c_{out}	$Somme$
0	0	0	0	0
0	0	1	0	1
0	0	2	0	2
0	1	0	0	1
0	1	1	0	2
0	1	2	1	0
0	2	0	0	2
0	2	1	1	0
0	2	2	1	1
1	0	0	0	1
1	0	1	0	2
1	0	2	1	0
1	1	0	0	2
1	1	1	1	0
1	1	2	1	1
1	2	0	1	0
1	2	1	1	1
1	2	2	1	2

Table 1: Table de vérité pour question Ib

En utilisant des signaux encodés en binaire, on obtient la table de vérité démontrée dans table 2.

Les 14 combinaisons (sur 32) restantes où $a_1 = a0 = 1$, ou $b_1 = b_0 = 1$, peuvent êre traitè comme des conditions *don't care*. Alors, les expressions de coût minimum sont:

$$c_{out} = a_0 \cdot b_1 + a_1 \cdot b_0 + a_1 \cdot b_1 + a_1 \cdot c_{in} + b_1 \cdot c_{in} + a_0 \cdot b_0 \cdot c_{in} \quad (4)$$

$$s_1 = a_0 \cdot b_0 \cdot \overline{c_{in}} + \overline{a_1} \cdot \overline{a_0} \cdot b_1 \cdot \overline{c_{in}} + a_1 \cdot \overline{b_1} \cdot \overline{b_0} \cdot \overline{c_{in}} + a_1 \cdot b_1 \cdot c_{in} + \overline{a_1} \cdot \overline{a_0} \cdot b_0 \cdot c_{in} + a_0 \cdot \overline{b_1} \cdot \overline{b_0} \cdot c_{in} \quad (5)$$

$$s_0 = a_1 \cdot b_1 \cdot \overline{c_{in}} + \overline{a_1} \cdot \overline{a_0} \cdot b_1 \cdot \overline{c_{in}} + a_0 \cdot \overline{b_1} \cdot \overline{b_0} \cdot \overline{c_{in}} + a_1 \cdot b_0 \cdot c_{in} + a_0 \cdot b_1 \cdot c_{in} + \overline{a1} \cdot \overline{a_0} \cdot \overline{b_1} \cdot \overline{b_0} \cdot c_{in} \quad (6)$$

	A		B		*Report*	*Somme*	
c_{in}	a_1	a_0	b_1	b_0	c_{out}	s_1	s_0
0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	0	1
0	0	1	0	1	0	1	0
0	0	1	1	0	1	0	0
0	1	0	0	0	0	1	0
0	1	0	0	1	1	0	0
0	1	0	1	0	1	0	1
1	0	0	0	0	0	0	1
1	0	0	0	1	0	1	0
1	0	0	1	0	1	0	0
1	0	1	0	0	0	1	0
1	0	1	0	1	1	0	0
1	0	1	1	0	1	0	1
1	1	0	0	0	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	0	1	1	0

Table 2: Table de vérité pour question Ib

Question II

Cette question se concerne avec le circuit de multiplicateur à rangée. La partie a est question 3.15 de votre manuel.

Partie a

Le chemin le plus long, qui cause le délai critique, est des entrées m_0 et m_1 à la sortie p_7, indiqué par le chemin à tiret sur la figure 1.

La propagation par le bloc A implique un délai d'une porte dans la porte logique ET démontré dans la figure 5.32b et deux délais pour produire le report de sortie de l'additionneur-complet. Puis, dans chacun des blocs B, C, D, E, F, G, et H, deux délais de plus sont nécessaire pour produire des signaux de reports de sortie dans les circuits représentés par la figure 5.32c. Par conséquent, le délai total le long du chemin critique est 17 délais de porte logique.

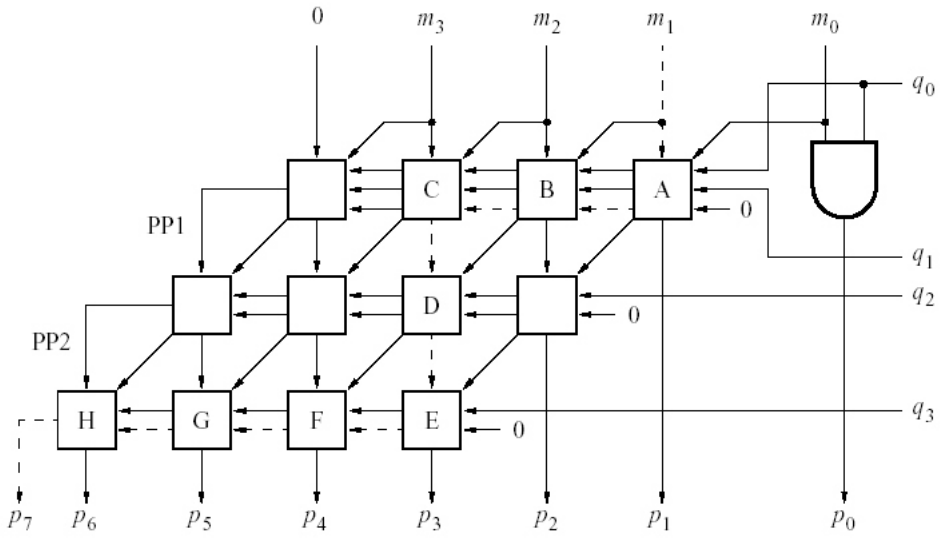

Figure 1: Multiplicateur à rangée

Partie b

La plus grande valeur d'un nombre de n-chiffre, si non signé dans la base B, est B^n-1. Cette valeur au carré nous donne le plus grand possible résultat de $B^{2n}-2B^n+1$. Cette valeur est plus petite que $B^{2n}-1$, et par conséquent peut être représentée en utilisant $2n$ chiffres.

Ainsi, la multiplication de deux nombres de n-chiffre rapportera toujours un produit représentable en $2n$ chiffres.

Question III

Cette question se concerne avec les tables de transition d'états. Les parties a, b et c sont questions 6.9, 6.10 et 6.23 de votre manuel.

Partie a

Pour comparer deux bits individuels, on laisse $k = w_1 \oplus w_2$. Alors, des tables d'état et de transition appropriées peuvent être comme démontré dans la table .
Finalement, les équations de conception sont:

$$Y_2 = \overline{k} \cdot y_1 + \overline{k} \cdot y_2$$

$$Y_1 = \overline{k} \cdot \overline{y_1} + \overline{k} \cdot y_2$$

Present état	*Prochain état*		*Rendement*	
	$k = 0$	$k = 1$	$k = 0$	$k = 1$
A	B	A	0	0
B	C	A	0	0
C	D	A	0	0
D	D	A	1	0

Présent état y_2y_1	*Prochain état* $k = 0$ Y_2Y_1	*Prochain état* $k = 1$ Y_2Y_1	*Rendement* $k = 0$	*Rendement* $k = 1$
A→ 00	01	00	0	0
B→ 01	10	00	0	0
C→ 10	11	00	0	0
D→ 11	11	00	1	0

Table 3: Tables d'état et de transition de la question IIIa

$$z = \overline{k} \cdot y_1 \cdot y_2$$

Partie b

L'entité *questionIIIb* est codée en VHDL comme ci-dessous:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY questionIIIb IS
PORT (i_clock : IN STD_LOGIC;
      i_resetb : IN STD_LOGIC;
      i_w1, i_w2 : IN STD_LOGIC;
      o_z : OUT STD_LOGIC);
END questionIIIb;

ARCHITECTURE Behavior OF questionIIIb IS
TYPE stateType IS (A, B, C, D);
SIGNAL int_state: stateType;
SIGNAL int_k: STD_LOGIC;
BEGIN
    int_k <= i_w1 XOR i_w2;
    PROCESS (i_resetb, i_clock)
    BEGIN
        IF (i_resetb = '0') THEN
            int_state <= A;
        ELSIF (i_clock'EVENT AND i_clock = '1') THEN
            CASE int_state IS
                WHEN A =>
                    IF int_k = '0' THEN
                        y <= B;
                    ELSE y <= A;
                    END IF;
                WHEN B =>
```

Present état	*Prochain état*		*Rendement*		
	$w = 0$	$w = 1$	z_2	z_1	z_0
A	A	B	0	0	0
B	B	C	0	0	1
C	C	D	0	1	0
D	D	E	0	1	1
E	E	F	1	0	0
F	F	A	1	0	1

Présent état $y_2y_1y_0$	*Prochain état*		*Rendement*		
	$w = 0$	$w = 1$			
	$Y_2Y_1Y_0$	$Y_2Y_1Y_0$	z_2	z_1	z_0
A→ 000	000	001	0	0	0
B→ 001	001	010	0	0	1
C→ 010	010	011	0	1	0
D→ 011	011	100	0	1	1
E→ 100	100	101	1	0	0
F→ 101	101	000	1	0	1

Table 4: Tables d'état et de transition de la question IIIc

```
                    IF int_k = '0' THEN
                        y <= C;
                    ELSE y <= A;
                    END IF;
                WHEN C =>
                    IF int_k = '0' THEN
                        y <= D;
                    ELSE y <= A;
                    END IF;
                WHEN D =>
                    IF int_k = '0' THEN
                        y <= D;
                    ELSE y <= A;
                    END IF;
            END CASE;
        END IF;
    END PROCESS;

    z <= '1' WHEN int_state = D and int_k = '0' ELSE
         '0';
END Behavior;
```

Partie c

Des tables d'état et de transition appropriées peuvent être comme démontré dans la table 4.
Finalement, les équations de conception sont:

$$Y_2 = \overline{y_0} \cdot y_2 + \overline{w} \cdot y_2 + w \cdot y_0 \cdot y_1$$

$$Y_1 = \overline{y_0} \cdot y_1 + \overline{w} \cdot y_1 + w \cdot y_0 \cdot \overline{y_1} \cdot \overline{y_2}$$

$$Y_0 = \overline{w} \cdot y_0 + w \cdot \overline{y_0}$$

$z_2 = y_2$

$z_1 = y_1$

$z_0 = y_0$

Question IV

Cette question se concerne avec les FSMs de type Mealy et Moore. Les schémas utilisés sont figures 6.43 et 6.47 dans votre manuel.

Le diagramme de synchronisation des FSMs est démontré dans la figure 2. On peut constater que la somme d'une machine de Mealy change avec les entrées, a et b, pendant que la somme d'une machine de Moore change une période d'horloge après le changement de la sortie de la machine de Mealy. Cela est causé par la bascule d'état utilisé dans la figure 8.47 dans votre manuel pour produire la variable de présent état y_1 et la sortie s.

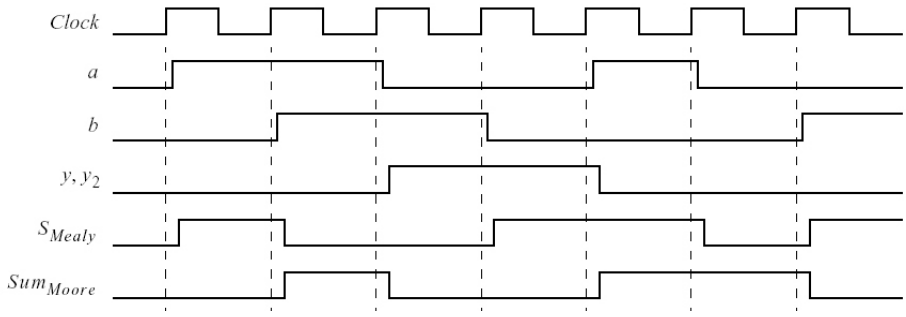

Figure 2: Diagramme de synchronisation

Question de Boni

Démontrez, en utilisant un diagramme de synchronisation, l'opération correcte d'une bascule de type D, montrée dans la figure 3 et présentée en classe.

Le diagramme de synchronisation qui démontre correctement l'opération d'une bascule de type D est présenté dans la figure 4. Notez que les étiquettes qui indiquent les états suivants:

- Quand l'horloge est haute, le verrou maître maintient ses valeurs de sortie, (*int_D* and *int_DBar*), tandis que le verrou esclave enregistre ses données à ses sorties (*Q* and *QBar*);

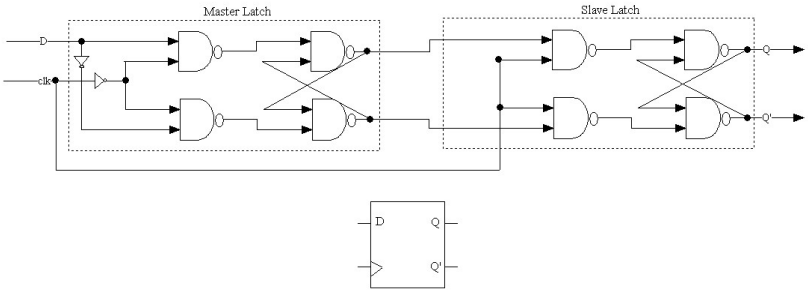

Figure 3: Bascule de type D en configuration maître-esclave

- `Quand l'horloge est basse, le verrou maître enregistre ses données à ses sorties, tandis que le verrou esclave maintient ses valeurs de sortie.`

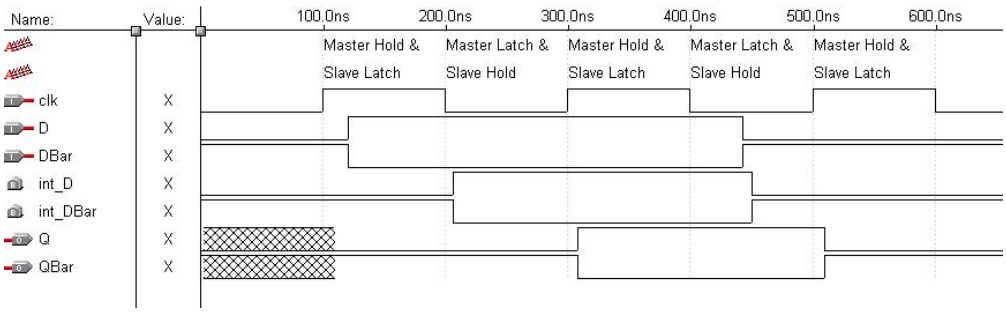

Figure 4: Diagramme de synchronisation d'une bascule de type D

`Alors, quand l'horloge est haute, les changements de l'entrèe (`*`D`*`) sont ignorés jusqu'au prochain bord de l'horloge, tandis que quand l'horloge est basse, les changements aux entrées du verrou esclave sont ignorés jusqu'au prochain bord de l'horloge. Le résultat est que les changements de l'entrée` *`D`* `sont seulement observé à la sortie` *`Q`* `à la transition de bas à haut de l'horloge (supposant des bascules déclenchées par front d'impulsion positif).`

Remerciements

Les réponses et figures liées aux questions du manuel sont tirées du manuel d'accompagnement de l'instructeur pour *Fundamentals of Digital Logic with VHDL Design* par Stephen Brown et Zvonko Vranesic.