ÉCOLE CENTRALELYON

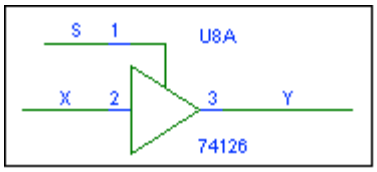
TRAVAUX DIRIGES N°7

I/ Etude d'un multiplieur

I.1. Ecrire la table de vérité puis l'équation logique d'une multiplication de 2 bits a0 et b0.
I.2. Détailler le principe de la multiplication sur un exemple (mots de 4 bits) : **2 x 5**
I.3 En considérant l'exemple 4 bits **15 x 15**, définir la taille nécessaire du mot de sortie.
I.4. Ecrire l'équation logique et arithmétique de la sortie en fonction des entrées.
I.5. Dessiner un schéma électrique simple à partir de portes et d'additionneurs 8 bits.

II/ Buffer 8 bits à sortie haute-impédance

Rappel du TD 5 : soit le composant suivant (74LS126 buffer 3 états, 1 bit) ci-dessous.

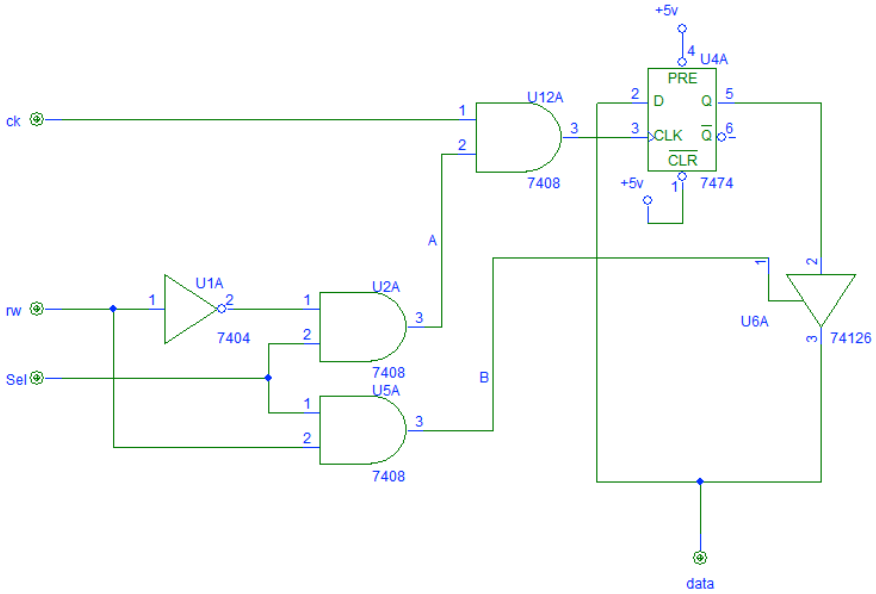


Y = X si S = 1
Y = Z si S = 0

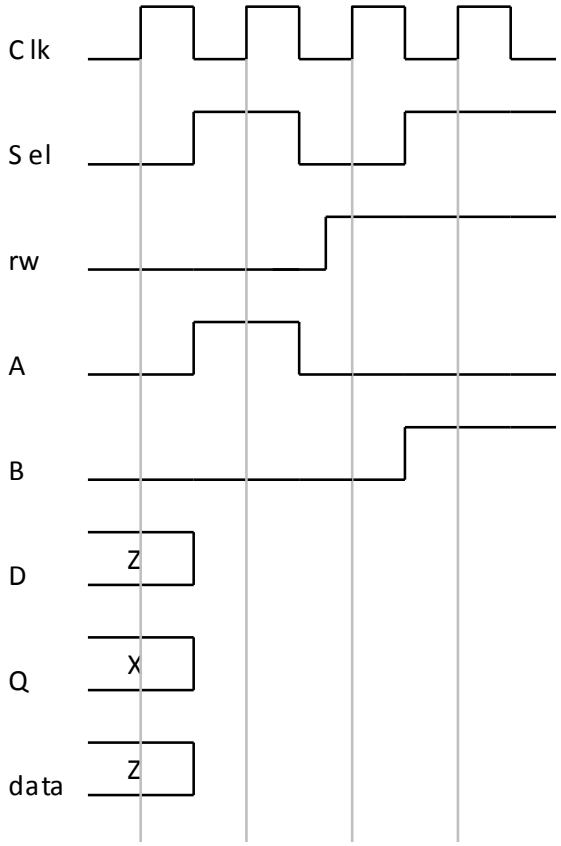
A partir de ce composant, proposer une architecture permettant de réaliser un buffer 8 bits dont les sorties peuvent être mises à l'état "haute-impédance".

III/ Point mémoire

Le schéma ci-dessous représente un point mémoire bidirectionnel utilisant une bascule D. Le signal de commande écriture / lecture est *rw*, le signal de sélection (mémoire active / inactive) est *sel*, l'horloge est *ck*, la donnée est *data*.



III.1. Rappeler le fonctionnement de la bascule D.
III.2. Donner les équations de A et B en fonction de *rw* et *sel*.
III.3. Expliquer le fonctionnement du schéma, et sur le chronogramme suivant, identifier les phases de lecture, d'écriture et de haute impédance et enfin compléter l'entrée D, la sortie Q et le signal data.



III.4. Ce schéma peut être modifié pour créer un point mémoire 8 bits. En considérant le schéma précédent comme un bloc ayant les entrées sorties Sel, rw, ck, data, dessiner le schéma de ce point mémoire 8 bits.

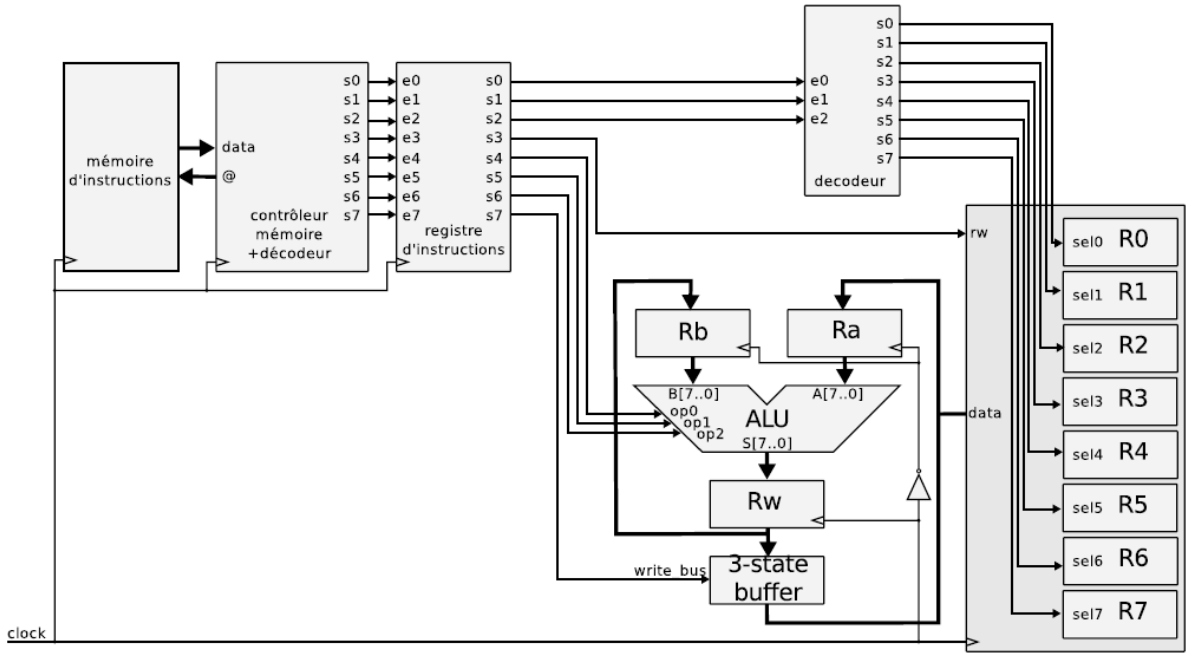
IV/ Décodeur 3 vers 8

En utilisant – entre autres – des portes logiques 74LS126, dessiner la structure interne d'un démultiplexeur 1 vers 2 à sorties haute impédance ayant cette table de vérité :

Entrée de sélection S	Entrée E	Sortie s0	Sortie s1
0	0	0	Z
0	1	1	Z
1	0	Z	0
1	1	Z	1

V/ Etude d'un chemin de données avec une UAL (ALU) 4 bits

Soit l'architecture suivante:



Rappels :
- L'unité de calcul peut être configurée en 5 opérations, selon cette table de vérité :

OP2	OP1	OP0	S
0	0	0	A + B
0	0	1	A – B
0	1	0	A x B
0	1	1	A
1	X	X	A / B

- Le Buffer 3 états fonctionne selon cette table de vérité :

E[7..0]	write_bus	bus[7..0]
Data	0	ZZZZZZZZ
Data	1	Data

où data est un mot 8 bits

L'entrée *rw* permet de lire ("1") ou écrire ("0") dans la mémoire de données.

Etude globale

V.1. Expliquer la structure et le fonctionnement de chaque bloc de ce schéma.
V.2. Identifier où se situent les mémoires de programme et de données.
V.3. Expliquer le fonctionnement général

Jeu d'instructions et programmation

Le cycle fetch permet de lire une instruction (code exécutable sur 8 bits) en mémoire et de la transférer dans le registre d'instruction. 256 instructions différentes sont donc théoriquement possibles. Pour simplifier l'exemple, le contrôleur mémoire se contente de

lire successivement les codes exécutables et de les transférer de la mémoire vers le registre d'instructions.

V.4. Détailler le mot de contrôle (issu du registre d'instructions 8 bits) en expliquant le rôle de chacun des bits.

V.5. Donner la succession de mots 8 bits (code exécutable) permettant de faire l'addition des cases mémoires (registres) 1 et 2, puis de placer le résultat dans la case mémoire 3.

En analysant tout ce que peut faire ce chemin de données, définir le jeu d'instructions assembleur (qui est plus simple à utiliser que les codes exécutables vu précédemment), en utilisant:
Ra comme nom du registre A
Rb comme nom du registre B
Rw comme nom du registre W
Rx comme nom du registre n°x (0 … 7)

V.6. Définir et expliquer une instruction de plus haut niveau permettant d'échanger les valeurs de 2 registres : SWAP(X,Y).

TRAVAIL EN AUTONOMIE

Simulations sous PSPICE

Etude d'une UAL (ALU) 4 bits

Définir l'architecture et simuler le fonctionnement d'un chemin de données composé d'une Unité Arithmétique et Logique (UAL, *ALU*) 4 bits (74181), et d'un registre (bascules D) 4 bits (74175).

L'opération à faire en simulation est : 2 x E – 1
(E étant l'entrée de donnée sur 4 bits, qui vaudra 3)

Extrait de datasheet du 74181 :
On notera qu'il faut configurer les entrées : M = 0, Cn = 1

MODE SELECT INPUTS				ACTIVE HIGH INPUTS AND OUTPUTS	
S_3	S_2	S_1	S_0	LOGIC (M=H)	ARITHMETIC(2) (M=L; C_n=H)
L	L	L	L	$\overline{A}$	A
L	L	L	H	$\overline{A + B}$	A + B
L	L	H	L	$\overline{A}B$	A + $\overline{B}$
L	L	H	H	logical 0	minus 1
L	H	L	L	$\overline{AB}$	A plus A$\overline{B}$
L	H	L	H	$\overline{B}$	(A + B) plus A$\overline{B}$
L	H	H	L	A ⊕ B	A minus B minus 1
L	H	H	H	A$\overline{B}$	A$\overline{B}$ minus 1
H	L	L	L	$\overline{A}$ + B	A plus AB
H	L	L	H	$\overline{A \oplus B}$	A plus B
H	L	H	L	B	(A + $\overline{B}$) plus AB
H	L	H	H	AB	AB minus 1
H	H	L	L	logical 1	A plus A(1)
H	H	L	H	A + $\overline{B}$	(A + B) plus A
H	H	H	L	A + B	(A + $\overline{B}$) plus A
H	H	H	H	A	A minus 1

Pin Names	Description
$\overline{A}0$–$\overline{A}3$	Operand Inputs (Active LOW)
$\overline{B}0$–$\overline{B}3$	Operand Inputs (Active LOW)
S0–S3	Function Select Inputs
M	Mode Control Input
C_n	Carry Input
$\overline{F}0$–$\overline{F}3$	Function Outputs (Active LOW)
A = B	Comparator Output
$\overline{G}$	Carry Generate Output (Active LOW)
$\overline{P}$	Carry Propagate Output (Active LOW)
C_{n+4}	Carry Output