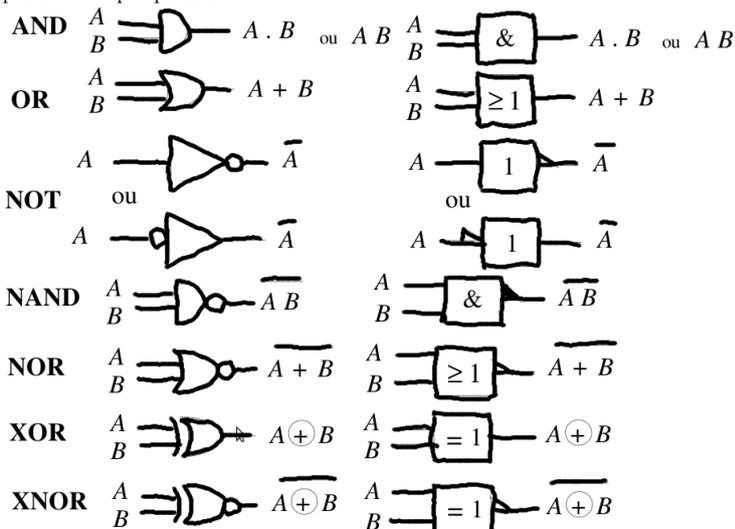


1 Fiche combinatoire - ADO

Mise en forme Babillon Damien avec les notes d'Elías Harrous et le glandage manifeste de Benoît Materrazi. ING1 2k15, ©, rédigé L^AT_EX.

1.1 Logique combinatoire

Principe de dualité : Une équation logique reste vraie si on remplace + par . et 0 par 1 et réciproquement



Identificateur général

Symbole	Description
&	Fonction ET
>= 1	Fonction OU
=1	Fonction OU Exclusif
=	Fonction Coïncidence
2k	Le nombre d'entrées activées doit être pair (pour activer la sortie)
2k+1	Le nombre d'entrées activées doit être impair (pour activer la sortie)
1	L'entrée unique doit être active (pour activer la sortie)
X/Y	Codeur, convertisseur (DEC/BCD, BIN/7 segments)
MUX	Multiplexeur
MUX ou DX	Démultiplexeur
Σ	Additionneur
P-Q	Soustracteur (Comparateur numérique)
CPG	Générateur de retenue anticipée
π	Multiplieur
COMP	Comparateur (amplitude analogique)
ALU	Unité arithmétique et logique
SRG m	Registre à décalage (m = nombre de bits)
CTR m	Compteur m bits (cycle de 2m états)
CTR DIV m	Compteur de cycle = m
ROM	Mémoire morte (Read Only Memory)
RAM	Mémoire vive (Random Access Memory)
FIFO	Mémoire vive file d'attente (First In First Out)

OU Exclusif (XOR) = $\overline{C}.D + C.\overline{D}$

Coïncidence (XNOR) = $\overline{C}.\overline{D} + C.D$

Codeur : Dispositif qui traduit les valeurs d'une entrée dans un code choisi.

Par exemple, un clavier de console ou de machine à écrire comporte m touches. Chaque touche, est affectée d'un numéro. L'opération de codage consiste à donner à chaque numéro un équivalent binaire

Dans sa version la plus générale, un codeur est un ensemble de circuits OU

Décodeur : Un décodeur réalise la fonction inverse (≡ duale) du codeur

La réalisation des décodeurs se fait à partir d'une matrice ET

Démultiplexeur : Il réalise l'opération duale du multiplexeur : il aiguille 1 donnée d'entrée sur 1 parmi 2ⁿ sorties (n entrées d'adresses)

Codage	N	SVA (BCD)	C2
+3	011	011	
+2	010	010	
+1	001	001	
0	000	000	
-0	100	-	
-1	101	111	
-2	110	110	
-3	111	101	
-4	-	100	

Multiplexage (ou sélecteur ou aiguilleur)

C'est un circuit réalisant un aiguillage (recopie) de l'une des entrées de données (par la commande des entrées d'adresse) vers une sortie unique. Il y a sélection d'une donnée parmi 2ⁿ (n entrées d'adresses)
 Soit $f_2(a, b) = a + b$
 $\Rightarrow f_2(a, b) = \overline{a} \cdot \overline{b} \cdot f_2(0, 0) + \overline{a} \cdot b \cdot f_2(0, 1) + a \cdot \overline{b} \cdot f_2(1, 1) + a \cdot b \cdot f_2(1, 0)$

1.2 Logique Séquentielle

Système logique combinatoire : La seule connaissance des entrées suffit à déterminer les sorties

Système logique séquentiel (≡ système logique combinatoire bouclé) : A l'instant discret n, une sortie s_jⁿ d'un système logique séquentiel dépend de ses entrées e₁ⁿ, ..., e_pⁿ mais aussi de l'état antérieur des sorties (s₁ⁿ⁻¹, ..., s_mⁿ⁻¹) qui peuvent être considérées comme des entrées secondaires, alors que les e₁ⁿ, ..., e_pⁿ sont appelées primaires (notion de mémoire, car les systèmes séquentiels sont bouclés ≡ récursifs). La seule connaissance des entrées (primaires) ne suffit pas à déterminer l'état des sorties

1.2.1 Les Bascules

Fonctionnement asynchrone : La sortie de la bascule change d'état uniquement en fonction des grandeurs d'entrée

- Le système est plus rapide que les systèmes synchrones
- Mais il présente des temps de propagation (≡ délais) difficiles à maîtriser

Fonctionnement synchrone : La prise en compte des entrées est conditionnée par une autorisation donnée par un signal d'horloge.

positive edge triggered = synchro sur front montant	latch > 0 = synchro sur niveau haut
negative edge triggered = synchro sur front descendant	latch < 0 = synchro sur niveau bas

S	R	Q _n	Fonction	Complémentarité
0	0	Q _{n-1}	Mémorisation	Q' = \overline{Q}
0	1	0	RESET (Mise à 0 de Q)	Q' = \overline{Q}
1	0	1	SET (Mise à 1 de Q)	Q' = \overline{Q}
1	1		Combinaison interdite car Q' != Q	Q' != \overline{Q}

Horloge	T	S	R	Q _n	Fonction
Inactive	0	X	X	Q _{n-1}	Mémorisation
Active	1	0	0	Q _{n-1}	Mémorisation
Active	1	0	1	0	RESET (Remise à 0 de Q)
Active	1	1	0	1	SET (Mise à 1 de Q)
Active	1	1	1		Interdit

T=0 : la sortie

ne change pas quelles que soient les entrées R et S (bascule non synchronisée)
 T = 1 : sa sortie respecte la table de fonctionnement de la bascule RS (asynchrone) avec les mêmes restrictions (bascule synchronisée)

Horloge	C	D	Q _n	Fonction
Inactive	0	X	Q _{n-1}	Mémorisation
Active	1	D		Recopie

La bascule D impose une restriction

pour le bon fonctionnement : Exemple : pour une bascule D latch > 0, D ne doit pas changer d'état pendant que C = 1

Horloge	T	Q _n	Fonction
Inactive	0	Q _{n-1}	Mémorisation
Active	1	\overline{Q}_{n-1}	Complémentation

Remarque : Si le signal de commande est périodique de période TH (fréquence f), alors le signal de sortie est également périodique mais de période 2T_H (fréquence $\frac{f}{2}$)

Horloge	H	J	K	Q _n	Fonction
Inactive	X	X	X	Q _{n-1}	Mémoire
Active	0	0	0	Q _{n-1}	Mémoire
Active	0	1	0		RESET (Remise à 0 de Q)
Active	1	0	1		SET (Mise à 1 de Q)
Active	1	1	1	\overline{Q}_{n-1}	Complémentation

Bascule JK = bascule RS avec : J = S, K = R et la combinaison J = K = 1 est non interdite et de type bascule T

Table des transitions (Horloge active)

Transition	Q _{n-1} → Q _n	J	K
0 → 0		0	X
0 → 1		1	X
1 → 1		X	0
1 → 0		X	1

Les Registres : Un registre est d'abord un ensemble de cases ou cellules mémoire capables de stocker une information (≡ un mot binaire). La position des cases mémoire entre elles est responsable de l'ordre des chiffres, c'est à dire de la structure de l'information. Dans le système binaire, une case mémoire est définie à l'aide d'une bascule. Un registre est donc un ensemble ordonné de bascules. Moyennant une interconnexion entre les cellules, un registre devient capable d'opérer une translation des chiffres (≡ bits) du nombre (≡ mot) initialement stocké.

1.2.2 Les Compteurs (généralisation de la notion de registre)

C'est un ensemble de bascules dont les sorties forment un mot binaire et qui compte dans une séquence donnée à chaque coup d'horloge. Il est représenté par un automate d'états fini encore appelé machines d'états : c'est l'horloge qui fait passer d'un état au suivant, contrairement au séquenceur dont les changements d'état sont soumis aussi à des actions et des transitions

Il existe deux types de compteurs

- Les compteurs asynchrones ou compteurs série → propagation en cascade de l'ordre de changement d'état des bascules (avec des retards!)
- L'horloge synchronise la 1ère bascule, dont la sortie va synchroniser la bascule suivante
- Les compteurs synchrones ou compteurs parallèles → Le signal d'horloge synchronise toutes les bascules simultanément

Un compteur binaire m bits compte de 0 à 2^{m-1}

N	Sorties				Entrées							
	A	B	C	D	J _A	K _A	J _B	K _B	J _C	K _C	J _D	K _D
0	0	0	0	0	1	X	0	X	0	X	0	X
8	1	0	0	0	X	0	1	X	0	X	0	X
12	1	1	0	0	X	0	X	0	1	X	0	X
14	1	1	1	0	X	1	X	0	X	0	1	X
7	0	1	1	1	1	X	X	1	X	0	X	0
11	1	0	1	1	X	0	1	X	X	1	X	0
13	1	1	0	1	X	1	X	0	1	X	X	1
6	0	1	1	0	0	X	X	1	X	0	1	X
3	0	0	1	1	1	X	0	X	X	1	X	0
9	1	0	0	1	X	1	1	X	0	X	X	1
4	0	1	0	0	1	X	X	1	1	X	0	X
10	1	0	1	0	X	1	1	X	X	1	1	X
5	0	1	0	1	0	X	X	1	1	X	X	1
2	0	0	1	0	0	X	0	X	X	1	1	X
1	0	0	0	1	0	X	0	X	0	X	X	1

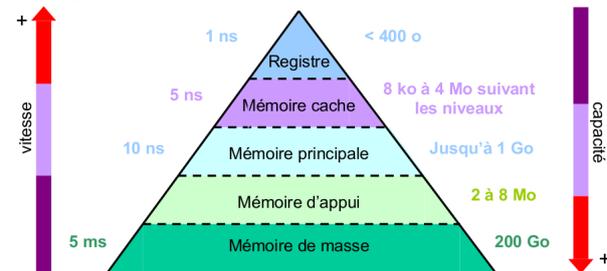
1.2.3 Exercice TD4.3

Synthétiser un compteur 2 bits synchrone avec des bascules JK positive edge triggered, qui compte selon le cycle suivant : Q₁Q₀ = 00 → 01 → 11 → 00 → ...

	Q ₁	Q ₀	J ₁	K ₁	J ₀	K ₀
Etat 1	0	0	0	X	1	X
↓						
Etat 2	0	1	1	X	X	0
↓						
Etat 3	1	1	X	1	X	1

A l'Etat 1, les J et K sont définis de telle sorte que l'on obtienne les Q_1Q_0 de l'Etat 2
 A l'Etat 2, les J et K sont définis de telle sorte que l'on obtienne les Q_1Q_0 de l'Etat 3
 A l'Etat 3, les J et K sont définis de telle sorte que l'on obtienne les Q_1Q_0 de l'Etat 1
 Ensuite, il faut établir les tables de Karnaugh de :
 - J_0K_0 en fonction de Q_0 et Q_1
 - J_1K_1 en fonction de Q_0 et Q_1
 On obtient $J_0 = 1, K_0 = Q_1, J_1 = Q_0$ et $K_1 = 1$

Transférer des données : Une mémoire RAM, reliée au processeur, a un temps de cycle T_c de 20 ns pour le premier accès et de 10 ns pour les trois accès suivants qui sont accélérés. Chaque accès récupère 8 octets. Quelle est la Bande Passante (en Octets/seconde) du transfert d'informations entre la mémoire et le processeur ?
 $4 \text{ accès avec } 8 \text{ octets à chaque accès pour un total de } 20+10+10+10 = 50 \text{ ns et } 8 \times 4 = 32. \frac{32}{50 \cdot 10^{-9}} = 640 \text{ Mo/s.}$
Mémoire et espace d'adressage
 1. capacité 13 fils d'adresses et 8 fils de données : $2^{13} \times 8 \text{ bits, } 2^{13} \text{ octets, } 2^{10} \cdot 2^3 = 2^3 \text{ ko} = 2^3 \times 8 = 64 \text{ kbits.}$
 2. espace adressable bus d'adresse de 16 fils et d'un bus de données de 8 bits : $2^{16} \text{ octets} \Rightarrow 2^6 \text{ ko} = 64 \text{ ko. } 2^{19} \text{ bits} \Rightarrow 512 \text{ kbits.}$
Le format des données : c'est le nombre de bits que l'on peut mémoriser par case mémoire. On dit aussi que c'est la largeur du mot mémorisable
Le temps d'accès : c'est le temps qui s'écoule entre l'instant où a été lancée une opération de lecture/écriture en mémoire et l'instant où la première information est disponible sur le bus de données
Le temps de cycle : il représente l'intervalle minimum qui doit séparer deux demandes successives de lecture ou d'écriture
Volatilité : elle caractérise la permanence des informations dans la mémoire. L'information stockée est volatile si elle risque d'être altérée par un défaut d'alimentation électrique et non volatile dans le cas contraire



1.3 Microprogrammation (Chemin des données)

ADD R1 (R2) et mettre le resultat dans R1

R1	R2	IR	MDR	MAR	PC	Y	Bus1	Bus2	micro-ins	commentaire
0xDD	0xAB	-	-	-	0x03	-	-	-	-	état initial
								0x03	PC out	(PC) → bus 2
								0x03	REPB	F = (PC) = 0x03
				0x03					MAR in	F = (PC) = 0x03
								0x04	INCRB	F = 0x04
					0x04				PC in	PC = 0x04
			0xFA						Lecture	0xFA → MDR
								0xFA	MDR out	B = 0xFA
								0xFA	REPB	F = 0xFA
		0xFA							IR in	0xFA → IR; décode
								0xAB	R2 out	B = 0xAB
								0xAB	REPB	F = 0xAB
				0xAB					MAR in	MAR = 0xAB
					0x27				Lecture	MDR = 0x27
								0x27	MDR out	B = 0x27
								0x27	REPB	F = 0x27
						0x27			Y in	A = 0x27
								0xDD	R1 out	B = 0xDD
								0x04	ADD	F = 0xDD + 0x27 = 0x04
0x04									R1 in	R1 = 0x04

1 1 1 1 1 | 1 1 1
 0 0 1 0 | 0 1 1 1 | 27
 + 1 1 0 1 | 1 1 0 1 | -23
 0 0 0 0 | 0 1 0 0 | 4
 sont égales).

donc calcul correct, pas de débordement (les dernières retenues)

1.4 Processeurs actuels

Un cycle représente le temps nécessaire pour exécuter 1 étage du pipeline
 Le débit d'exécution correspond au nbre d'instructions par cycle

$$\equiv \frac{\text{Nombre d'étages du pipeline}}{\text{Nombre de cycles nécessaire l'exécution}}$$
max théorique = 1 instruction/cycle
 Latence = durée d'exécution, temps de traitement d'une instruction
 Latence totale = temps de traitement de tous les étages du pipeline
 Temps de cycle = Temps de traitement d'1 étage du pipeline
 = Latence de l'étage + Latence de latch (registre de sortie)

1.5 Architecture de von Neumann

Etapes d'exécution d'une instruction

- Chargement de l'instruction $\begin{cases} IP & \text{adresse} \\ IR & \text{contenu de l'instruction} \end{cases}$
- Décodage de l'instruction + lecture des registre
- UAL : Unité Arithmétique & Logique
- Accès mémoire (lecture ou écriture d'une donnée en mémoire) si nécessaire
- Ecriture aux registres (écriture du résultat dans un registre)

Instruction ASM	Commentaire	IP	AX	BX	Flag Z	SP	Stack
0 Etat Initial		0100	0000	0000	0	FFFE	00 00 00 00 00 00
1 MOV AX, 01 00 _h	Ecrit 01 00 _h	0103	0100	0000	0	FFFE	00 00 00 00 00 00
2 MOV BX, 03 04 _h	Ecrit 03 04 _h dans le registre BX; (BX)=03 04	0106	0100	0304	0	FFFE	00 00 00 00 00 00
3 Boucle : ADD AL, 1	Ajoute 1 à l'octet de poids faible de AX noté AL : (AL)=(AL)+1	0108	0101	0304	0	FFFE	00 00 00 00 00 00
4 CMP AL, 2	Compare (AL) à 2; place le bit de Flag Z à 1 en cas d'égalité de la comparaison	010A	0101	0304	0	FFFE	00 00 00 00 00 00
5 JNE Boucle	Saut à l'étiquette Boucle si le bit Z = 0 (s'il n'y a pas égalité)	0106	0101	0304	0	FFFE	00 00 00 00 00 00
6 ADD AL, 1		0108	0102	0304	0	FFFE	00 00 00 00 00 00
7 CMP AL, 2		010A	0102	0304	1	FFFE	00 00 00 00 00 00
8 JNE Boucle		010C	0102	0304	1	FFFE	00 00 00 00 00 00
9 PUSH AX	Empile le contenu de AX dans la pile (STACK) : (AX)→STACK	010D	0102	0304	1	FFFC	00 00 01 02 00 00
10 PUSH BX		010E	0102	0304	1	FFFA	00 00 01 02 03 04

1.7 Codage

Nombre à virgule flottante

Norme	Décomposition	Signe	Exposant (entier)	Mantisse
IEEE754	Simple Précision (32 bits)	1	8	23
	Double Précision (64 bits)	1	11	52

Nombre normalisé :
 $\text{nombre} = (-1)^{\text{signe}} * 1, \text{mantisse} * 2^{(\text{exposant} - 127)}$
 $10 = (-1)^0 \cdot 1.25 \cdot 2^3$
 $s(1) = 0$
 $\text{exp}(8) = (130)_{10} = 1000 \ 0010$
 $0.25 \times 2 = 0.5$
 $0.5 \times 2 = 1$
 $m(23) = (0.25)_{10} = 010 \ 0000 \ 0000 \ 0000 \ 0000 \ 0000$
 $\text{sem} = 0100 \ 0001 \ 0010 \ 0000 \ 0000 \ 0000 \ 0000 \ 0000 = 41 \ 20 \ 00 \ 00$
 Addition en IEEE754 simple précision de 1/10 et 1/10.
 a. Ramener les 2 nombres au même exposant
 b. on restaure le bit de poids fort
 c. on effectue l'addition ou la soustraction des valeurs absolues
 d. on renormalise le résultat

$0.1 = (1 + 0.6) \times 2^{123-127} = 0011 \ 1101 \ 1100 \ 1100 \ 1100 \ 1100 \ 1100 \ 1100 =$
 $0D \ CC \ CC \ CC$
 $1 \ 001 \ 1001 \ 1001 \ 1001 \ 1001 \ 100$
 $1.100 \ 1100 \ 1100 \ 1100 \ 1100 \ 1100.2^{123-127}$
 $1.100 \ 1100 \ 1100 \ 1100 \ 1100 \ 1100.2^{123-127}$
 $11.001 \ 1001 \ 1001 \ 1001 \ 1001 \ 1000.2^{123-127}$ On perd le dernier bit
 $1.100 \ 1100 \ 1100 \ 1100 \ 1100 \ 1100.2^{124-127}$

$s(1) = 0$
 $\text{exp}(8) = (124)_{10} = 0111 \ 1100$
 $m(23) = 100 \ 1100 \ 1100 \ 1100 \ 1100 \ 1100$
 $\text{sem} = 0011 \ 1110 \ 0100 \ 1100 \ 1100 \ 1100 \ 1100 \ 1100 = 3E \ 4C \ CC \ CC$
 Multiplication en IEEE754 simple précision de -18 par 10.
 a. calculer le signe $s_r = s_a \oplus s_b$; exposant $e_r = e_a + e_b - \text{biais}$
 b. on restaure le bit de poids fort
 c. on effectue la multiplication des valeurs absolues
 d. éventuellement, arrondir, ajuster l'exposant et on renormalise le résultat
 $-18 = 1100 \ 0001 \ 1001 \ 0000 \ 0000 \ 0000 \ 0000 \ 0000 = C1 \ 90 \ 00 \ 00$
 $10 = 0100 \ 0001 \ 0010 \ 0000 \ 0000 \ 0000 \ 0000 \ 0000 = 41 \ 20 \ 00 \ 00$
 a. $s_r(1) = 1, e_r = e_x + e_y - 127, (127)_{10} = 0111 \ 1111 \rightarrow (-127)_{10} = 1000 \ 0001$

$1 \ 0000 \ 011$
 $1000 \ 0011$
 $+ 1000 \ 0010$
 $+ 1000 \ 0001$
 $e_r = 1000 \ 0110 = 128 + 4 + 2 = 134$
 $m_r = m_x \times m_y \ b. (1 + m_x)(1 + m_y)$
 1.001
 $\times 1.01$
 1001
 $0 \ 000$ $m_z = 011 \ 0100 \ 0000 \ 0000 \ 0000 \ 0000$
 $10 \ 01$
 $m_z = 1.0 \ 1101$
 $\text{sem}(r) = 1100 \ 0011 \ 0011 \ 0100 \ 0000 \ 0000 \ 0000 \ 0000 = C3 \ 34 \ 00 \ 00$
 Intervalle des entiers naturels représentables en BCD sur 16 bits : $2^{16}-1$
 Intervalle des entiers relatifs représentables en Ca2 sur 16 bits : -2^{15} à $2^{15}-1$