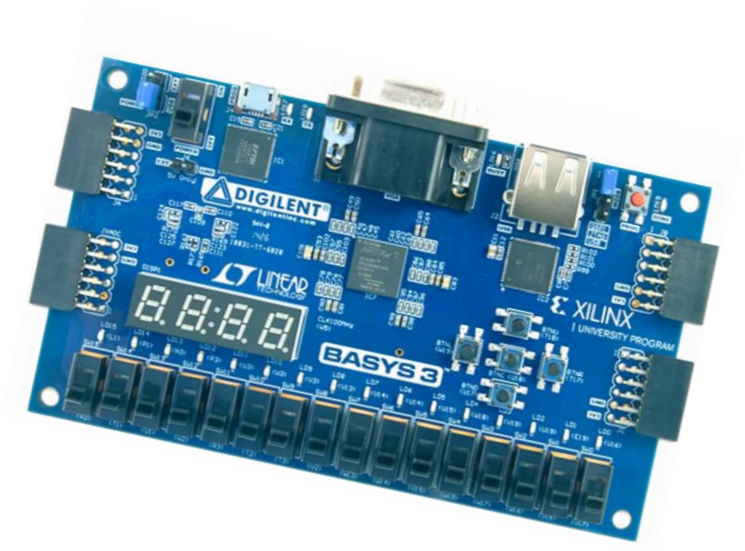
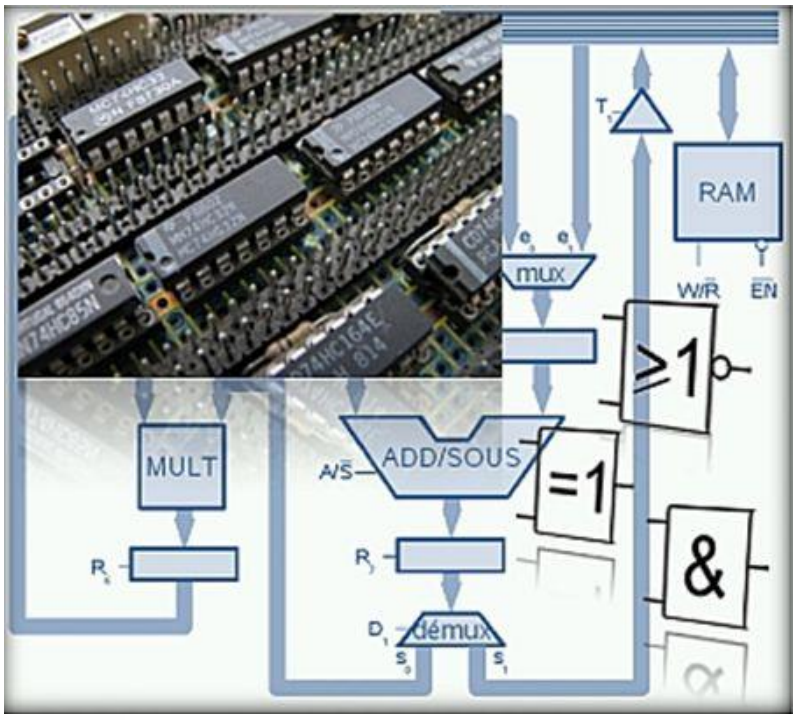
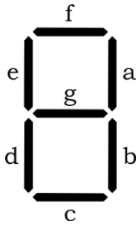
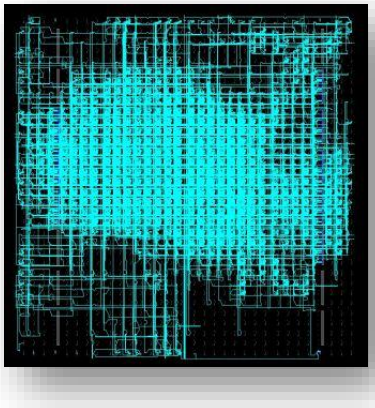
ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE



Electronique numérique
Circuits et architectures logiques
2025-2026 – 1A ELEC







M.Denoual

ENSICAEN

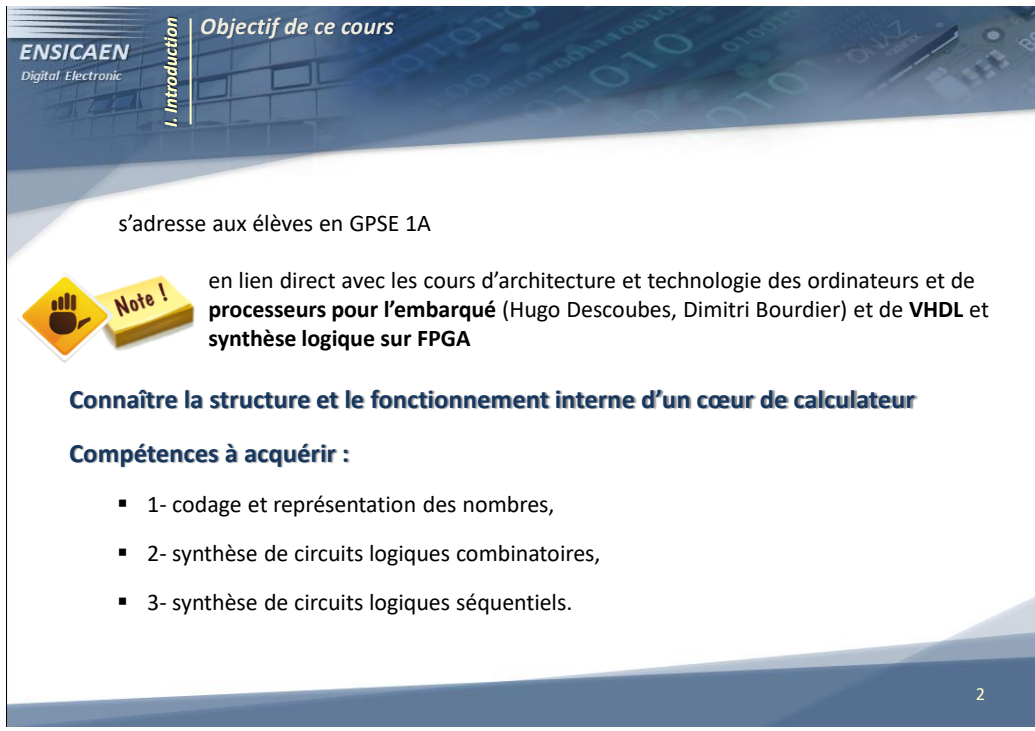
6 boulevard Maréchal Juin • CS 45053 • F • 14050 Caen Cedex 4

Tél. +33 (0)2 31 45 27 50

Fax +33 (0)2 31 45 27 60


ENSICAEN
Digital Electronic
I. Introduction
Contexte de l'électronique numérique
M. Denoual
1

1


ENSICAEN
Digital Electronic
I. Introduction
Objectif de ce cours
s'adresse aux élèves en GPSE 1A
Note !
en lien direct avec les cours d'architecture et technologie des ordinateurs et de **processeurs pour l'embarqué** (Hugo Descoubes, Dimitri Bourdier) et de **VHDL** et **synthèse logique sur FPGA**
Connaître la structure et le fonctionnement interne d'un cœur de calculateur
Compétences à acquérir :
1- codage et représentation des nombres,
2- synthèse de circuits logiques combinatoires,
3- synthèse de circuits logiques séquentiels.
2

2

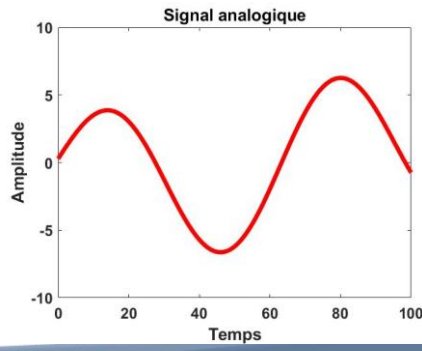
ENSICAEN
Digital Electronic

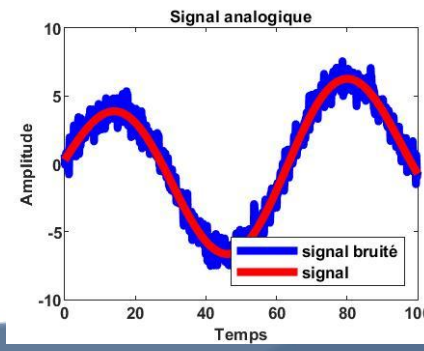
I. Introduction

Contexte de l'électronique numérique

Distinction électronique analogique/numérique

En électronique analogique, une information est par une grandeur électrique -tension ou courant- qui est connue de manière . Cette information peut prendre une dont la connaissance est limitée par le des composants et des fonctions analogiques utilisées.





3

ENSICAEN
Digital Electronic

I. Introduction

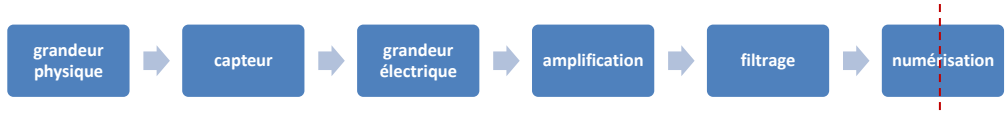
Contexte de l'électronique numérique

Distinction électronique analogique/numérique

- Les domaines d'application de l'électronique analogique ont tendance à se réduire au profit de l'électronique numérique.

- Les 3 domaines restants d'applications de l'électronique analogique sont:

  – 1. le conditionnement de signaux issus de capteurs (amplification et filtrage). L'objectif est alors de dégager un signal électrique souvent faible du bruit de façon à pouvoir l'utiliser ou le traiter.



Remarques :
- Toutes les grandeurs physiques sont analogiques
- Dans certains capteurs récents, on peut trouver des électroniques de numérisation directe sur puce qui repoussent encore le domaine de l'analogique.

4

4

– 2. communication radio-fréquence (RF) ou électronique haute-fréquence

– 3. électronique de puissance

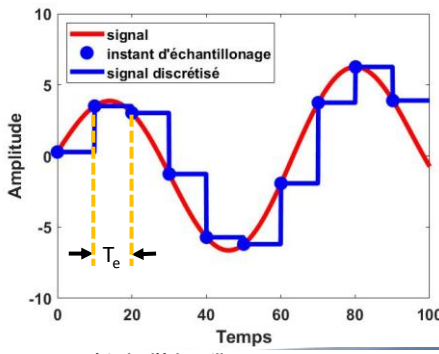
Remarques :

- On trouve également de l'électronique analogique dans les systèmes automatiques pour l'asservissement ou la régulation.
Mais la tendance générale en automatique est le développement à partir d'électronique numérique. Cette tendance est soutenue par les capacités croissant sans cesse des composants numériques (vitesse, capacité de calcul) et par la diminution de leur coût.

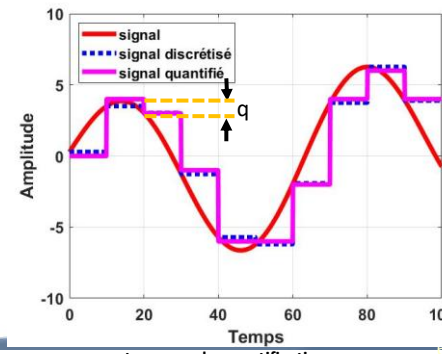
- L'électronique analogique peut présenter un intérêt pour des dispositifs peu complexes et faible coût.

5

- En électronique numérique ou électronique digitale (franglais), les signaux présentent des dans le temps et
- L'information est représentée au moyen d'un nombre limité de valeurs grâce à une ou plusieurs grandeurs binaires appelées bits.



T_e : période d'échantillonnage



q : quantum, pas de quantification

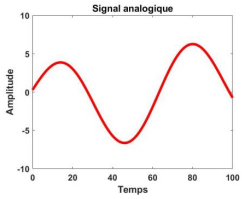
6

ENSICAEN
Digital Electronic

I. Introduction

Contexte de l'électronique numérique

Distinction électronique analogique/numérique

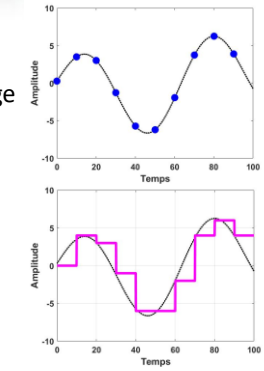


continue dans le temps ⇨ instant d'échantillonnage

discrétisation

infinité de valeur ⇨ nb de valeurs fini

quantification



Remarque :

contrainte sur les instants d'échantillonnages (théorème de Shannon-Nyquist)
Il faut que la **fréquence d'échantillonnage** (inverse de l'intervalle entre deux instants d'échantillonnage) soit supérieure à 2 fois la fréquence maximum du signal analogique.

7

7

ENSICAEN
Digital Electronic

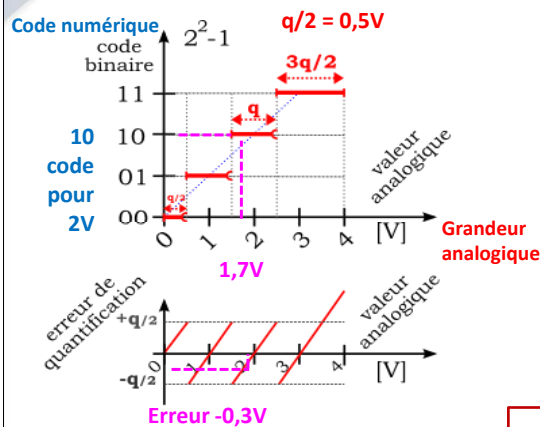
I. Introduction

Contexte de l'électronique numérique

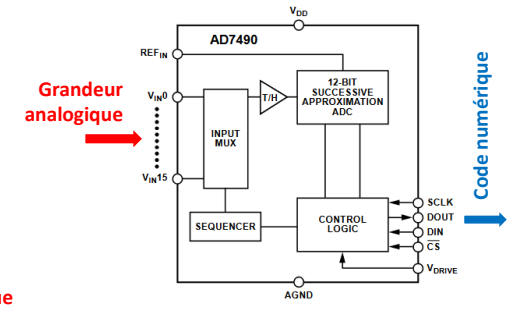
Distinction électronique analogique/numérique

AD7490 : convertisseur 16 canaux, 1MSPS, 12-bit

représentation linéaire centrée



FUNCTIONAL BLOCK DIAGRAM



*Gamme : [0; REF_{IN}] ou [0;2*REF_{IN}]*

Quantum : $q = \frac{gamme\ analogique}{2^N}$

8

8

ENSICAEN Digital Electronic | I. Introduction

Contexte de l'électronique numérique

Distinction électronique analogique/numérique

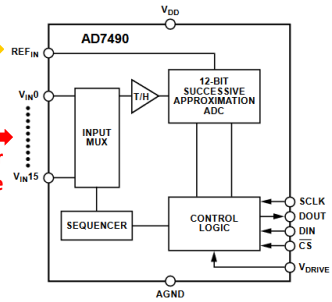
Quantum : $q = \frac{\text{gamme entrée analogique}}{2^N}$

Référence pour la gamme de tension d'entrée analogique

Grandeur analogique

AD7490 : convertisseur 16 canaux, 1MSPS, 12-bit

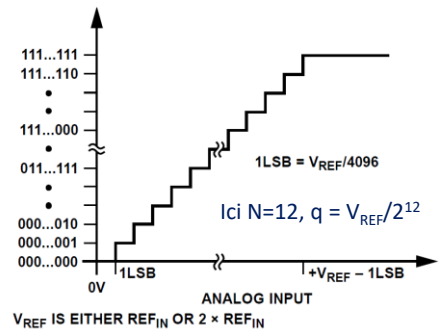
FUNCTIONAL BLOCK DIAGRAM



Code numérique

*Gamme : [0; REF_{IN}] ou [0;2*REF_{IN}]*

ADC TRANSFER FUNCTION



LSB : Least Significant Bit. Poids du bit de poids faible (quantum, pas de quantification).

ENSICAEN Digital Electronic | I. Introduction

Contexte de l'électronique numérique

Signaux numériques

bits : binary digit (chiffre binaire)

Ces bits prennent les valeurs :

valeur	contexte
vrai/faux	logique
0/1	logique booléenne
0V/5V	aspect électrique
L/H	*datasheet*
5V/0V	logique négative
12V/-12V	transmission série
...	

TRUTH TABLE

I_1	I_2	O_1
L	L	L
H	L	H
L	H	H
H	H	L

Notes

1. H = HIGH state (the more positive voltage)
   L = LOW state (the less positive voltage)

table de vérité dans une datasheet de composant logique discret

$^1/_4$ of device shown

$J = \overline{A + B}$

Logical "1" = HIGH

Logical "0" = LOW

All inputs protected by standard CMOS protection circuit.

relation entre valeur logique et niveau électrique dans une datasheet de composant logique discret

ENSICAEN
Digital Electronic

I. Introduction

Contexte de l'électronique numérique

Signaux numériques

Exemples de signaux numériques

- bouton poussoir (touche d'un clavier):
  BP=1, bouton enfoncé, interrupteur fermé
  BP=0, bouton relâché, interrupteur ouvert
- lampe : L=1, lampe allumée / L=0, lampe éteinte
- pixel d'une image (télévision numérique)
  - diffusion : bits

Smart Watch et autres petits écran format image RGB565 16 bits

Question : *combien de couleur peut-on coder en format diffusion ?*

12

12

ENSICAEN
Digital Electronic

I. Introduction

Contexte de l'électronique numérique

Distinction électronique analogique/numérique

Questions *:*

Combien de valeurs peut-on représenter avec N de bits ?

Quelle est l'erreur de quantification maximum en fonction du quantum?

On suppose une gamme de tension de [0-4V] et 2 bits de représentation, combien valent le quantum (pas de quantification) et l'erreur maximum de représentation ?

Application numérique pour N=10 bits et une gamme de tension [0-5V]

13

13

ENSICAEN
Digital Electronic

I. Introduction

Contexte de l'électronique numérique

Signaux numériques

Exemples de signaux numériques (suite)

- CD audio, MP3 :
  - bits
  - fréquence échantillonnage :

Question : *taille d'une image FHD 1920x1080 pixels format BMP ?*

Question : *quelle taille mémoire pour 1 heure de musique, stéréo non compressée ?*

- Signal numérisé sortant d'un capteur : 8 à 12 bits. Exemple, capteur de température DS18P20 : 10 bits

- Code ASCII : bits (caractères)
(*American Standard Code for Information Interchange*)

14

14

ENSICAEN
Digital Electronic

I. Introduction

Contexte de l'électronique numérique

Signaux numériques : code ASCII

Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII
0	0	NUL	32	20		64	40	@	96	60	`
1	1	SOH	33	21	!	65	41	A	97	61	a
2	2	STX	34	22	"	66	42	B	98	62	b
3	3	ETX	35	23	#	67	43	C	99	63	c
4	4	EOT	36	24	$	68	44	D	100	64	d
5	5	ENQ	37	25	%	69	45	E	101	65	e
6	6	ACK	38	26	&	70	46	F	102	66	f
7	7	BEL	39	27	'	71	47	G	103	67	g
8	8	BS	40	28	(	72	48	H	104	68	h
9	9	HT	41	29	)	73	49	I	105	69	i
10	A	LF	42	2A	*	74	4A	J	106	6A	j
11	B	VT	43	2B	+	75	4B	K	107	6B	k
12	C	FF	44	2C	,	76	4C	L	108	6C	l
13	D	CR	45	2D	-	77	4D	M	109	6D	m
14	E	SOH	46	2E	.	78	4E	N	110	6E	n
15	F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	|
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	□

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	Ç	160	A0	á	192	C0	└	224	E0	α
129	81	ü	161	A1	í	193	C1	┴	225	E1	ß
130	82	é	162	A2	ó	194	C2	┬	226	E2	Γ
131	83	â	163	A3	ú	195	C3	├	227	E3	π
132	84	ä	164	A4	ñ	196	C4	─	228	E4	Σ
133	85	à	165	A5	Ñ	197	C5	┼	229	E5	σ
134	86	å	166	A6	ª	198	C6	╞	230	E6	µ
135	87	ç	167	A7	º	199	C7	╟	231	E7	τ
136	88	ê	168	A8	¿	200	C8	╚	232	E8	Φ
137	89	ë	169	A9	⌐	201	C9	╔	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	╩	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	╦	235	EB	δ
140	8C	î	172	AC	¼	204	CC	╠	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	═	237	ED	φ
142	8E	Ä	174	AE	«	206	CE	╬	238	EE	ε
143	8F	Å	175	AF	»	207	CF	╧	239	EF	∩
144	90	É	176	B0	░	208	D0	╨	240	F0	≡
145	91	æ	177	B1	▒	209	D1	╤	241	F1	±
146	92	Æ	178	B2	▓	210	D2	╥	242	F2	≥
147	93	ô	179	B3	│	211	D3	╙	243	F3	≤
148	94	ö	180	B4	┤	212	D4	╘	244	F4	⌠
149	95	ò	181	B5	╡	213	D5	╒	245	F5	⌡
150	96	û	182	B6	╢	214	D6	╓	246	F6	÷
151	97	ù	183	B7	╖	215	D7	╫	247	F7	≈
152	98	ÿ	184	B8	╕	216	D8	╪	248	F8	°
153	99	Ö	185	B9	╣	217	D9	┘	249	F9	∙
154	9A	Ü	186	BA	║	218	DA	┌	250	FA	·
155	9B	¢	187	BB	╗	219	DB	█	251	FB	√
156	9C	£	188	BC	╝	220	DC	▄	252	FC	ⁿ
157	9D	¥	189	BD	╜	221	DD	▌	253	FD	²
158	9E	₧	190	BE	╛	222	DE	▐	254	FE	■
159	9F	ƒ	191	BF	┐	223	DF	▀	255	FF	

représenté sur 8 bits (256 caractères). Exemple : $6B_h$ représente le caractère « k »

15

15

ENSICAEN Digital Electronic | I. Introduction

Contexte de l'électronique numérique

électronique numérique

- L'électronique numérique est particulièrement adaptée aux calculs et aux tests.
- Ces calculs et tests sont la base des algorithmes de traitement par exemple en traitement numérique du signal (TNS) pour le filtrage, pour le cryptage ou la commande de procédé.

Rmq ! ***Remarque :*** *en électronique analogique, la seule façon de faire des calculs est d'utiliser des amplificateurs opérationnels (d'où le nom).*

- Les signaux numériques présentent en outre la possibilité d'être compressés (codage MP3, zip) et stockés longtemps.

- D'un point de vue transmission et également stockage et traitement, les informations numériques présentent une meilleure immunité au bruit. Cette caractéristique est très utile pour les communications pour réduire le taux d'erreur et peut être accrue par un codage (code correcteur d'erreur).

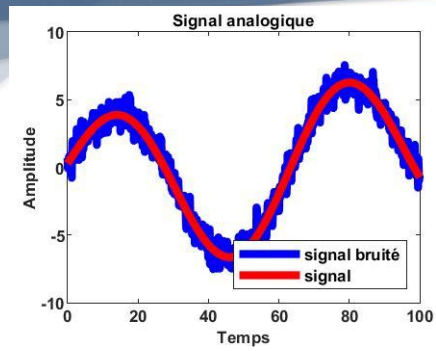
16

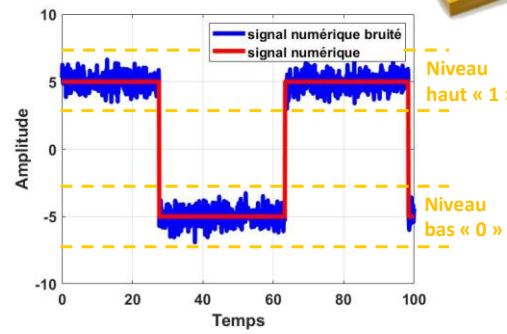
ENSICAEN Digital Electronic | I. Introduction

Contexte de l'électronique numérique

électronique numérique

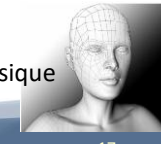
Remarque : moins d'information ici qu'à gauche Rmq !





- La confidentialité des transmissions peut être obtenue par cryptage (on peut également crypter en analogique mais il y a moins de possibilités).

- Champs des informations créées par l'homme : image virtuelle, jeux, musique numérique, films 3D…



17

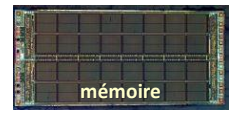
ENSICAEN
Digital Electronic

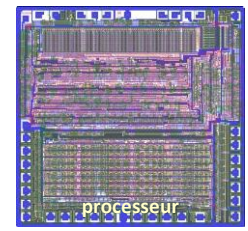
I. Introduction

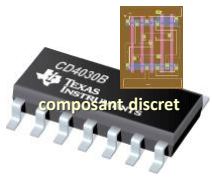
Contexte de l'électronique numérique

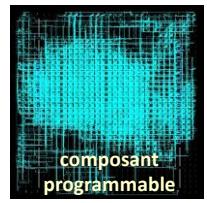
électronique numérique

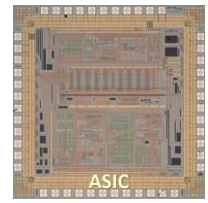
- **Mémoire** : stockage
- **Microprocesseurs** : exécution d'instruction de programme
- **Composants logiques** : fonctions spécifiques (ASIC, FPGA, PLD)











18

ENSICAEN
Digital Electronic

I. Introduction

A l'issue de cette première partie

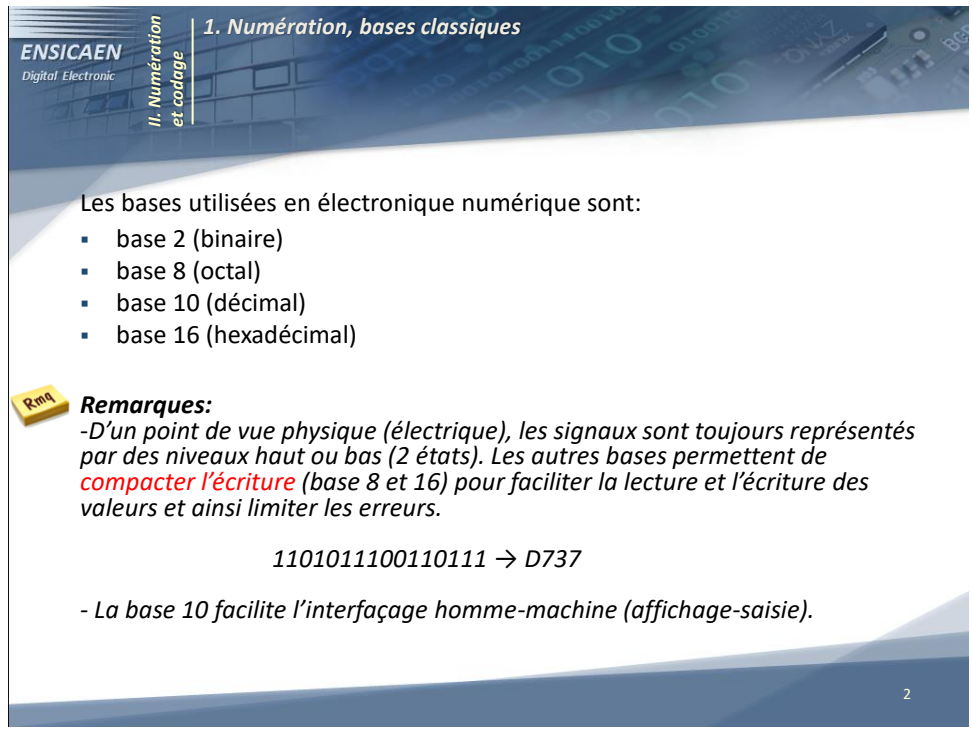
- Qu'est-ce qu'un signal discrétisé ?
- Qu'est-ce qu'un signal quantifié ?
- Quels intérêts à utiliser des signaux numériques plutôt qu'analogiques ?
- Quelles limites à l'utilisation des signaux numériques ?

19

19


ENSICAEN
Digital Electronic
II. Numération et Codage
Représentation des grandeurs au cœur des architectures
M. Denoual
1

1


ENSICAEN
Digital Electronic
II. Numération et codage
1. Numération, bases classiques
Les bases utilisées en électronique numérique sont:
base 2 (binaire)
base 8 (octal)
base 10 (décimal)
base 16 (hexadécimal)
Rmq
Remarques:
-D'un point de vue physique (électrique), les signaux sont toujours représentés par des niveaux haut ou bas (2 états). Les autres bases permettent de compacter l'écriture (base 8 et 16) pour faciliter la lecture et l'écriture des valeurs et ainsi limiter les erreurs.
1101011100110111 → D737
- La base 10 facilite l'interfaçage homme-machine (affichage-saisie).
2

2

Rmq **Remarques:**

- *Toute base s'écrit dans son propre système de numération*

- *Un décalage à gauche un nombre par sa base*

- *Un décalage à droite un nombre par sa base*

? ***Question :*** *combien faut-il de chiffres n dans une base B pour représenter un nombre décimal de N chiffre ?*

Application numérique: base 2 et N=2.

On attribue le poids de chaque chiffre.

Soit un nombre N= $(a_n\ a_{n-1}\ ....\ a_1\ a_0\ ,\ a_{-1}\ a_{-2}\ ...)_B$
alors la valeur en base 10 correspondant est :
$(N)_{10} = a_n B^n + a_{n-1} B^{n-1} + ... a_1 B^1 + a_0 B^0 + a_{-1} B^{-1} + a_{-2} B^{-2} + ...$

Exemples :

Divisions successives par la base

On traite différemment les parties entières et fractionnaires.
Soit un nombre N et sa représentation en base B sous la forme :

$$(a_n\ a_{n-1}\ ....\ a_1\ a_0\ ,\ a_{-1}\ a_{-2}\ ...)_B$$

E[N] Frac[N]

La partie entière, E[N], peut se mettre sous la forme :

$E[N] = a_n B^n + a_{n-1} B^{n-1} + ... + a_2 B^2 + a_1 B^1 + a_0 B^0$

en factorisant par la base B, on obtient :

$E[N] = B(a_n B^{n-1} + a_{n-1} B^{n-2} + ... + a_2 B^1 + a_1 B^0) + a_0$

1ère division donne a_0
2è division donne a_1
3è division donne a_2
...

en répétant l'opération :

$E[N] = B(B(a_n B^{n-1} + a_{n-1} B^{n-2} + ... + a_2 B^0) + a_1) + a_0$

On fait ainsi apparaître que les a_k peuvent être obtenus par division successive de E[N] par B.

5

5

Multiplications successives par la base

(cont.)

La partie fractionnaire, Frac[N], peut se mettre sous la forme :

$Frac[N] = a_{-1} B^{-1} + a_{-2} B^{-2} + ... + a_{-m} B^{-m}$

en multipliant par la base, on obtient :

$B.Frac[N] = a_{-1} + a_{-2} B^{-1} + ... + a_{-m} B^{-m+1}$

1ère multiplication donne a_{-1}
2è division donne a_{-2}
3è division donne a_{-3}
...

en répétant l'opération :

$B^2.Frac[N] = B\ a_{-1} + (a_{-2} + ... + a_{-m} B^{-m+2})$

On fait ainsi apparaître que les a_{-k} peuvent être obtenus par multiplications successives par B.

6

6

ENSICAEN
Digital Electronic

II. Numération et codage

2. Conversions, passage d'une base à l'autre

c. Conversion entre base 2^k-aire

- de la base 2 vers base 2^k avec k>1

regroupement des bits par k

base 2 vers base 8 : regroupement par 3 des bits

base 2 vers base 16 : regroupement par 4 des bits

- des bases 2^k avec k>1 vers la base 2

remplacement des chiffres de la base 2^k par leurs expressions dans la base 2

- d'une base 2^k (k>1) vers une autre base 2^k (k>1)

le plus simple est de passer par l'intermédiaire de la base 2 (développement puis regroupement)

exemples

Base 2 : 010011100011

Base 8 : 2 3 4 3

Base 2 : 010011100011

Base 16 : 4 E 3

7

7

ENSICAEN
Digital Electronic

II. Numération et codage

3. Codages à partir des deux chiffres 0 et 1

1010 

D'un point de vue de l'électronique numérique, les données réellement manipulées sont des **niveaux haut ou bas**, c'est-à-dire des 1 ou des 0 binaires.

A partir de ces deux chiffres, il existe plusieurs possibilités pour le codage des grandeurs ou signaux. Ces possibilités sont choisies en fonction de leur adéquation à une fonction (affichage, capteur position), de leur précision ou dynamique de représentation ou bien de leur simplicité d'implémentation (réduction de l'électronique associée).

Par la suite, on distinguera les codes pondérés des codes non pondérés.

Définitions :

un code est pondéré si chaque chiffre est affecté d'un poids.

un code est continu si les mots binaires consécutifs sont adjacents, c'est-à-dire qu'ils ne diffèrent que d'un bit.

un code est cyclique s'il est continu et si le premier et le dernier mots sont adjacents.

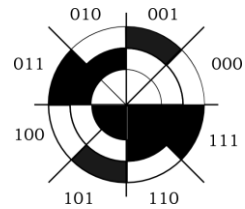
8

8

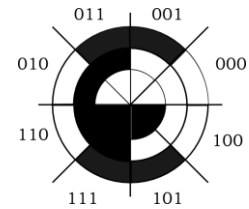
Code Gray

Le code Gray ou binaire réfléchi est cyclique par construction.

Valeur	code Gray		
0	0	0	0
1	0	0	1
2	0	1	1
3	0	1	0
4	1	1	0
5	1	1	1
6	1	0	1
7	1	0	0



binaire classique



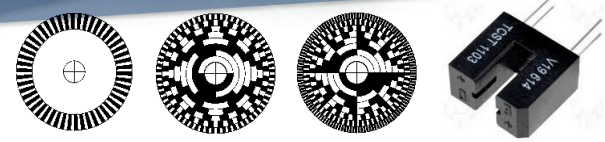
code Gray

Utilisé dans les codeurs de position optique
(une erreur de lecture -bit flou- ne conduit à une incertitude que sur deux valeurs successives)
Utilisé dans certaines machines d'état pour réduire la taille des blocs logiques

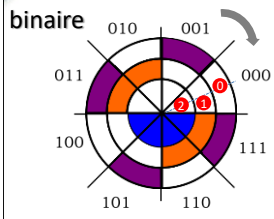
9

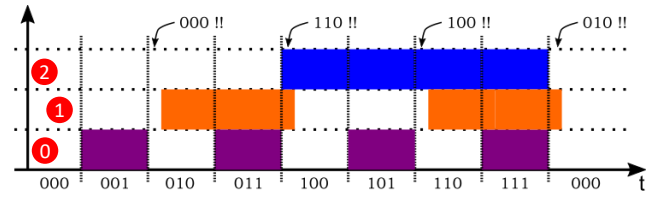
9

Application code Gray

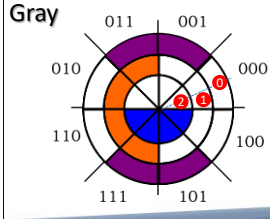


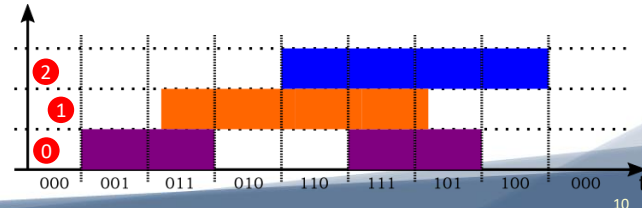
binaire





Gray





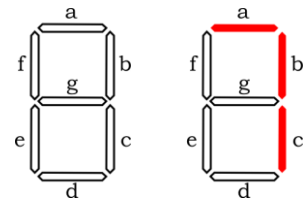
10

10

Code 7 segments

Codage utilisé pour les afficheurs 7 segments

chiffre à afficher	code 7 segments						
	a	b	c	d	e	f	g
0	1	1	1	1	1	1	0
1	0	1	1	0	0	0	0
2	.	.	.	.	.	.	.
3	.	.	.	.	.	.	.
4	.	.	.	.	.	.	.
5	.	.	.	.	.	.	.
6	.	.	.	.	.	.	.
7	1	1	1	0	0	0	0
8	.	.	.	.	.	.	.
9	.	.	.	.	.	.	.



Code BCD (*Binary Coded Decimal*)

Aussi appelé code DCB (Décimal codé binaire)
Dans ce codage, chaque chiffre d'un nombre décimal est codé individuellement en binaire.

Exemple : 2 3 1

Rmq ***Remarque :*** *on ne peut pas faire de calcul facilement avec ce code.*

3. Codages à partir des deux chiffres 0 et 1

b. Autres codes

	Codes DCB pondérés								Codes DCB non pondérés			
N_{10}	DCBN 8421	biquinaire 10 86420	biquinaire 50 43210	7421	MBQ 5 421	5211	Aiken 2421	753 -6	XS - 3	décimal Johnson	Gray DCB	2 parmi 5
0	0000	01 00001	01 00001	0000	0 000	0000	0000	000 0	0011	00000	0000	11000
1	0001	10 00001	01 00010	0001	0 001	0001	0001	100 1	0100	00001	0010	00011
2	0010	01 00010	01 00100	0010	0 010	0100	0010	011 1	0101	00011	0110	00101
3	0011	10 00010	01 01000	0011	0 011	0101	0011	001 0	0110	00111	0111	00110
4	0100	01 00100	01 10000	0100	0 100	0111	1010	101 1	0111	01111	0101	01001
5	0101	10 00100	10 00001	0101	1 000	1000	0101	010 0	1000	11111	1101	01010
6	0110	01 01000	10 00010	0110	1 001	1010	1100	110 1	1001	11110	1111	01100
7	0111	10 01000	10 00100	1000	1 010	1011	1101	100 0	1010	11100	1110	10001
8	1000	01 10000	10 01000	1001	1 011	1110	1110	011 0	1011	11000	1010	10010
9	1001	10 10000	10 10000	1010	1 100	1111	1111	111 1	1100	10000	1000	10100
auto complémentaires						oui	oui	oui	oui			
cyclique										oui	oui	
détection d'erreur		oui	oui									oui

13

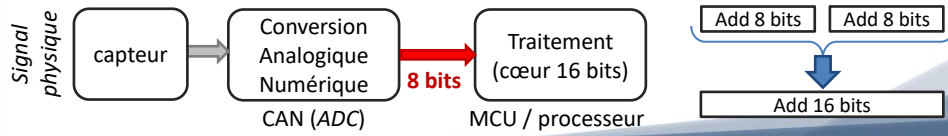
3. Codages à partir des deux chiffres 0 et 1

c. Codage pour le calcul

Problématique du codage pour le calcul

Caractéristiques requises de la représentation des nombres pour le calcul

- **bijectif** (chaque valeur n'a qu'une représentation)
- **extensible** (CAN 8 bits -> processeur 16 bits par instruction élémentaire)
- **simplicité des opérateurs de calculs (ALU)**
- **opérateurs cascadables** (2 additionneurs 8 bits pour former une additionneur 16 bits)
- **même représentation des nombres positifs en signé ou non**



14

binaire naturel (base 2)

sur N bits

$$x = b_{N-1}\,2^{N-1} + b_{N-2}\,2^{N-2} + \dots + b_2\,2^2 + b_1\,2^1 + b_0\,2^0$$

valeurs

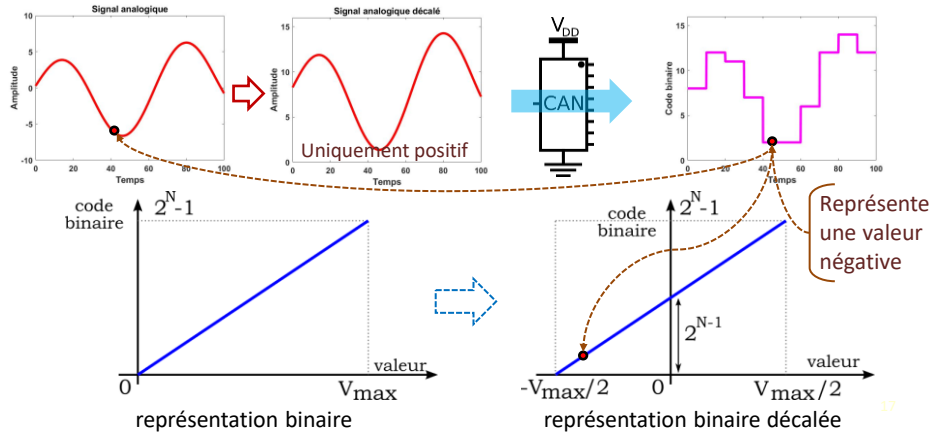
nombre entre [0;]

Propriétés : bijectif, extensible, opérateurs cascadables

15

binaire décalé

Ce type de codage vient des convertisseurs analogique/numérique (CAN), à cause de l'alimentation dissymétrique des CANs.



17

ENSICAEN
Digital Electronic

II. Numération et codage

3. Codages à partir des deux chiffres 0 et 1

c. Codage pour le calcul
Complément à 2

c'est LE codage utilisé dans les circuits et architectures logiques pour le calcul avec des nombres signés

Complément à 2

sur N bits

$$x = -b_{N-1}\,2^{N-1} + b_{N-2}\,2^{N-2} + \dots + b_2\,2^2 + b_1\,2^1 + b_0\,2^0$$

valeurs

nombre entre [;]

à retenir !

Exemples :

Propriétés : code bijectif, extensible, même représentation des nombres positifs en signé ou non, conduit à des structures d'opérateurs simples et cascadables.

important !

Règle d'extension (pour passer d'une représentation sur N bits à une représentation sur N+1 bits) : simplement par **recopie du bit de poids fort**.

nombre positif commence par : nombre négatif commence par :

20

20

ENSICAEN
Digital Electronic

II. Numération et codage

3. Codages à partir des deux chiffres 0 et 1

c. Codage pour le calcul
Complément à 2 (cont.)

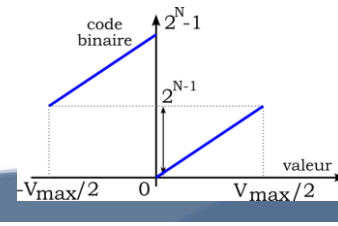
Rmq

Remarque : *tant que les bits de poids forts successifs sont identiques, on peut les supprimer pour ne garder de le dernier.*

Pour obtenir l'expression de la valeur négative d'un nombre, on procède en trois étapes comme suit :

à retenir !

1- expression de la valeur absolue du nombre (binaire naturel)
2- complément à 1 de chacun des bits
3- ajout d'un bit à '1' tout à droite



21

21

Exemples (1/2)

Exemples (2/2) -1563

1- expression de la valeur absolue du nombre (binaire naturel)

1563

1563	16
11	97 | 16
	1 | 6

1563 → 61B

0110 0001 1011

2- complément à 1 de chacun des bits

0110 0001 1011

1001 1110 0100

3- ajout d'un bit à '1' tout à droite

1001 1110 0100 +1

1001 1110 0101

Calcul de l'erreur de quantification. Vérification qu'elle est inférieure à l'erreur max (q/2)

ENSICAEN
Digital Electronic

II. Numération et codage

3. Codages à partir des deux chiffres 0 et 1

c. Codage pour le calcul
Complément à 2 (cont.)

Démonstration de cette construction en trois étapes

24

24

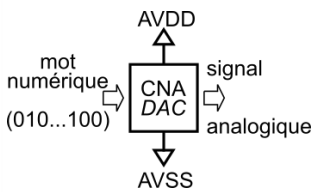
ENSICAEN
Digital Electronic

II. Numération et codage

3. Codages à partir des deux chiffres 0 et 1

Résumé représentation entiers non-signés/signés

Exemple d'un composant convertisseur numérique analogique (CNA, *DAC Digital Analog Converter*)



Data Sheet AD5724/AD5734/AD5754

Ideal Output Voltage to Input Code Relationship—AD5754

Table 7. Bipolar Output, Offset Binary Coding

Digital Input				Analog Output		
MSB			LSB	±5 V Output Range	±10 V Output Range	±10.8 V Output Range
1111	1111	1111	1111	+2 × REFIN × (32,767/32,768)	+4 × REFIN × (32,767/32,768)	+4.32 × REFIN × (32,767/32,768)
1111	1111	1111	1110	+2 × REFIN × (32,766/32,768)	+4 × REFIN × (32,766/32,768)	+4.32 × REFIN × (32,766/32,768)
…	…	…	…	…	…	…
1000	0000	0000	0001	+2 × REFIN × (1/32,768)	+4 × REFIN × (1/32,768)	+4.32 × REFIN × (1/32,768)
1000	0000	0000	0000	0 V	0 V	0 V
0111	1111	1111	1111	−2 × REFIN × (1/32,768)	−4 × REFIN × (1/32,768)	−4.32 × REFIN × (32,766/32,768)
…	…	…	…	…	…	…
0000	0000	0000	0001	−2 × REFIN × (32,767/32,768)	−4 × REFIN × (32,767/32,768)	−4.32 × REFIN × (32,767/32,768)
0000	0000	0000	0000	−2 × REFIN × (32,768/32,768)	−4 × REFIN × (32,768/32,768)	−4.32 × REFIN × (32,768/32,768)

Table 8. Bipolar Output, Twos Complement Coding

Digital Input				Analog Output		
MSB			LSB	±5 V Output Range	±10 V Output Range	±10.8 V Output Range
0111	1111	1111	1111	+2 × REFIN × (32,767/32,768)	+4 × REFIN × (32,767/32,768)	+4.32 × REFIN × (32,767/32,768)
0111	1111	1111	1110	+2 × REFIN × (32,766/32,768)	+4 × REFIN × (32,766/32,768)	+4.32 × REFIN × (32,766/32,768)
…	…	…	…	…	…	…
0000	0000	0000	0001	+2 × REFIN × (1/32,768)	+4 × REFIN × (1/32,768)	+4.32 × REFIN × (1/32,768)
0000	0000	0000	0000	0 V	0 V	0 V
1111	1111	1111	1111	−2 × REFIN × (1/32,768)	−4 × REFIN × (1/32,768)	−4.32 × REFIN × (1/32,768)
…	…	…	…	…	…	…
1000	0000	0000	0001	−2 × REFIN × (32,767/32,768)	−4 × REFIN × (32,767/32,768)	−4.32 × REFIN × (32,767/32,768)
1000	0000	0000	0000	−2 × REFIN × (32,768/32,768)	−4 × REFIN × (32,768/32,768)	−4.32 × REFIN × (32,768/32,768)

Table 9. Unipolar Output, Straight Binary Coding

Digital Input				Analog Output		
MSB			LSB	+5 V Output Range	+10 V Output Range	+10.8 V Output Range
1111	1111	1111	1111	+2 × REFIN × (65,535/65,536)	+4 × REFIN × (65,535/65,536)	+4.32 × REFIN × (65,535/65,536)
1111	1111	1111	1110	+2 × REFIN × (65,534/65,536)	+4 × REFIN × (65,534/65,536)	+4.32 × REFIN × (65,534/65,536)
…	…	…	…	…	…	…
1000	0000	0000	0001	+2 × REFIN × (32,769/65,536)	+4 × REFIN × (32,769/65,536)	+4.32 × REFIN × (32,769/65,536)
1000	0000	0000	0000	+2 × REFIN × (32,768/65,536)	+4 × REFIN × (32,768/65,536)	+4.32 × REFIN × (32,768/65,536)
0111	1111	1111	1111	+2 × REFIN × (32,767/65,536)	+4 × REFIN × (32,767/65,536)	+4.32 × REFIN × (32,767/65,536)
…	…	…	…	…	…	…
0000	0000	0000	0001	+2 × REFIN × (1/65,536)	+4 × REFIN × (1/65,536)	+4.32 × REFIN × (1/65,536)
0000	0000	0000	0000	0 V	0 V	0 V

25

3. Codages à partir des deux chiffres 0 et 1

c. Codage pour le calcul
Remarques générales

Remarques : décalages arithmétiques et logiques

- *un décalage à droite correspond à une division par 2*

exemple :

- *un décalage à gauche correspond à une multiplication par 2*

exemple:

Attention au débordement

- *Différence entre*

LSR (Logic Shift Right) décalage à droite logique (introduction d'un 0)
et ASR (Arithmetic Shift Right) (introduction du bit de signe)

- *Décalages spéciaux sur certains processeurs/microcontrôleurs*

par exemple PIC18 : décalage circulaire (RRNCF (rotate right f))

26

3. Codages à partir des deux chiffres 0 et 1

d. Représentations des nombres non entiers

Représentation à virgule fixe en $Q_{m,k}$

$$b_{N-1}\ b_{N-2}\ \ldots.\ b_k\ ,b_{k-1}\ \ldots.b_2\ b_1\ b_0$$

$$(2^{N-1}\ \ 2^{N-2}\ldots\ldots 2^k\ \ 2^{k-1}\ \ldots.\ \cdot 2^2\ 2^1\ 2^0\ ).\ 2^{-k}$$ k décalages

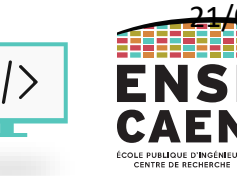
$$2^{N-1-k}\ \ 2^{N-2-k}\ldots\ldots 2^0\ \ 2^{-1}\ \ldots.\ \cdot 2^{2-k}\ 2^{1-k}\ 2^{-k}$$

Alors la valeur représentée appartient à l'intervalle :

Le quantum (pas de quantification) est :

***Remarque** : la virgule n'a pas d'existence matérielle. Les opérateurs arithmétiques traitent les nombres à virgule fixe comme des entiers (suite de 0 et de 1).*

27



ENSICAEN
Digital Electronic

II. Numération et codage

3. Codages à partir des deux chiffres 0 et 1

d. Représentations des nombres non entiers

Représentation à virgule fixe en $Q_{m,k}$ et complément à 2 **nombres non entiers signés**

On n'ajoute pas 1

Exemple : -12,25

12,25 : 0000 1100 0100

Complément à 1

1111 0011 1011

+1

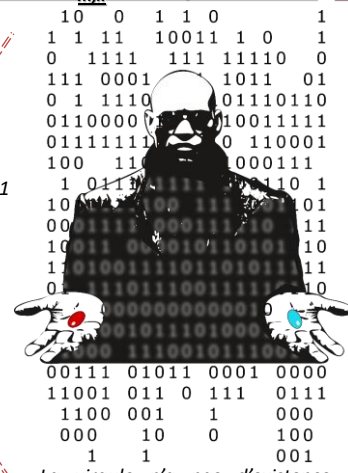
1111 0100 1011

1 0100 1011

-16 +4 +0,5 +0,125 +0,0625

-11,3125

FAUX



La virgule n'a pas d'existence matérielle. ***Dans la machine, il n'y a que des 0 et des 1.***

On ajoute un bit à 1 à droite de la représentation

Représentation $Q_{8,4}$

12,25 : 0000 1100 0100

Complément à 1

1111 0011 1011

+1 à droite

1111 0011 1100

1 0011 1100

-16 +2 +1 +0,5 +0,25

-12,25 ✓

28

28

ENSICAEN
Digital Electronic

II. Numération et codage

3. Codages à partir des deux chiffres 0 et 1

d. Représentations des nombres non entiers

Obtenir la représentation à virgule fixe en $Q_{m,k}$

Méthode 1 :

Si négatif : représentation de la valeur absolue puis complément à 2.

Pour les nbs positifs et la valeur absolue, obtention de la partie entière par divisions successives, obtention de la partie fractionnaire par multiplications successives.

Méthode 2 :

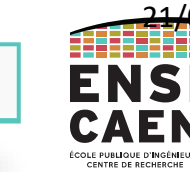
Multiplication du nombre à représenter par 2^k.

Représentation de l'entier obtenu sur m+k bits.

k décalages à droite de la représentation.

29

29



ENSICAEN
Digital Electronic

II. Numération et codage

3. Codages à partir des deux chiffres 0 et 1

d. Représentations des nombres non entiers

Exemples

30

30

ENSICAEN
Digital Electronic

II. Numération et codage

3. Codages à partir des deux chiffres 0 et 1

d. Représentations des nombres non entiers

Exemples conversion en base 2 (**16 bits** virgule fixe **Q12,4**) de -621,17.

nombre négatif, conversion en 3 étapes

Méthode 1

1- On représente la valeur absolue du nombre soit 621.17

partie entière div. succ. partie fractionnaire mult. succ.

621	16			0,17	0,72	
13	38	16	621→26D$_h$	x 16	x 16	0,17→0,2B$_h$
	6	2	621→001001101101	2,72	11,52	arrondi 0,17→0,3$_h$
						0,17→0,0011

2- Complément à « 1 » de chacun des bits 0010 0110 1101 0011

1101 1001 0010 1100

3- On ajoute un bit à « 1 » tout à droite +1

1101 1001 0010 1101

L'erreur est introduite lors de l'arrondi : 0,0011 → 0,1875

erreur = |0,1875-0,17|=0,0175 < erreur$_{max}$ = q/2 = 2^{-5} = 0.03125

31

31

Exemples conversion en base 2 (**16 bits** virgule fixe **Q12,4**) de -621,17.

Multiplication du nombre à représenter par 2^k ici 2^4 Méthode 2

$$-621{,}17 * 2^4 = -9938{,}72$$

Représentation de l'entier obtenu sur m+k bits, ici 16 bits. L'entier est négatif, on commence par représenter la valeur absolue du nombre.

$9938{,}72 \rightarrow$ *arrondi* 9939

Erreur de conversion introduite ici (arrondi)

9939	16	
3	621	16
	13	38	16
		6	2

$9939 \rightarrow 26D3$

0010 0110 1101 0011
1101 1001 0010 1100 Compl à 1
1101 1001 0010 1101 +1

k décalages à droite de la représentation.

1101 1001 0010 1101
Partie entière ⬆ Partie fractionnaire

Représentation à virgule flottante

Une valeur est représentée par une mantisse et un exposant.

$$x = \pm m.b^e$$

b : base; m : mantisse, e : exposant.

Question : *pourquoi virgule flottante ?*

Par la suite, on ne considère que l'étude en base 2 (b=2).

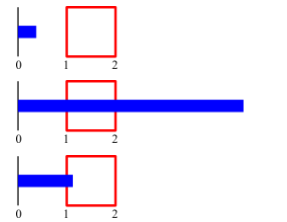
$$x = \pm m.2^e$$

On adapte le zoom

Représentation à virgule flottante (cont.)

Avec cette écriture, on peut avoir plusieurs solutions pour représenter une même valeur. Alors, on fixe une contrainte sur la mantisse de façon à avoir une écriture unique pour la valeur.

$$1 \leq |m| < 2$$



La mantisse est représentée en virgule fixe.
On utilise un bit pour le signe.

Finalement, le nombre en virgule flottante s'écrit :

$$x = (-1)^S.1,F.2^E$$

Le 1 de la partie entière n'a pas besoin d'être stocké. Il s'agit d'un 1 implicite.

Représentation à virgule flottante (cont.)

Norme IEEE 754 (1985)

C'est la représentation standard pour l'arithmétique en virgule flottante, utilisée par la plupart des systèmes pour améliorer la compatibilité entre les architectures et la portabilité des applications.

$$x = (-1)^S.1,F.2^{E-B}$$

B biais ou décalage
1+mantisse
(E≠0)

Exposant biaisé : l'exposant est représenté par un entier auquel on soustrait un biais B.

Exemple de la représentation IEEE 754 sur 32 bits (*simple precision*) :

exposant E sur 8 bits, F sur 23 bits : $x = (-1)^S.1,F.2^{E-127}$

$$S \quad E_7E_6E_5E_4E_3E_2E_1E_0 \quad F_{22}F_{21}F_{20}....F_2F_1F_0$$

Représentation IEEE 754 sur 64 bits (*double precision*), exposant E sur 11 bits.

Représentation à virgule flottante (cont.)

Norme IEEE 754 (1985) exemple

-44 828 721,152

1- mise en forme : signe puissance de 2 et mantisse

38

Représentation à virgule flottante (cont.)

Norme IEEE 754 (1985) exemple

A=-40 468,48

1- mise en forme : signe puissance de 2 et mantisse $A=(-1)^1\times1{,}235\times2^{15}$

$E-127 = 15 \rightarrow E = 142 \rightarrow 1000\ 1110$

0,235	0,76	0,16	0,56	0,96	0,36
x 16	x 16	x 16	x 16	x 16	x 16
3,76	12,16	2,56	8,96	15,36	5,76
0011	1100	0010	1000	1111	0101

011

	S	E7 E0	F22 F0
A	**1**	**1000 1110**	**0011 1100 0010 1000 1111 011**

39

Représentation à virgule flottante (cont.)

Norme IEEE 754 (cont.)

Nombres dénormalisés : en plus de la représentation normalisée avec le 1 implicite, la norme définit une représentation alternative lorsque l'exposant vaut 0. Pour représenter les valeurs très prochse de 0.

$$x = (-1)^S.0,F.2^{-126}$$

Exposant = 0
Mantisse ≠0

Cette dénormalisation permet de représenter le 0 lorsque F=0.

Valeur la plus petite dénormalisée : $0,0000....01.2^{1-B}$

Continuité avec normalisé : dénormalisé : $0,11111....111.2^{1-B}$
normalisé: $1,00000..0000.2^{1-B}$

$(-1)^S.0,11..11.2^{-126}$
$(-1)^S.1,00..00.2^{1-127}$

Représentations spéciales :

infini : représenté par un exposant maximum et F=0. Le bit de signe distingue + ∞ et -∞.

NaN : Not a Number, correspondant au résultat de calcul incorrect. Représenté par un exposant maximal et F non nul.

40

40

Représentation à virgule flottante (cont.)

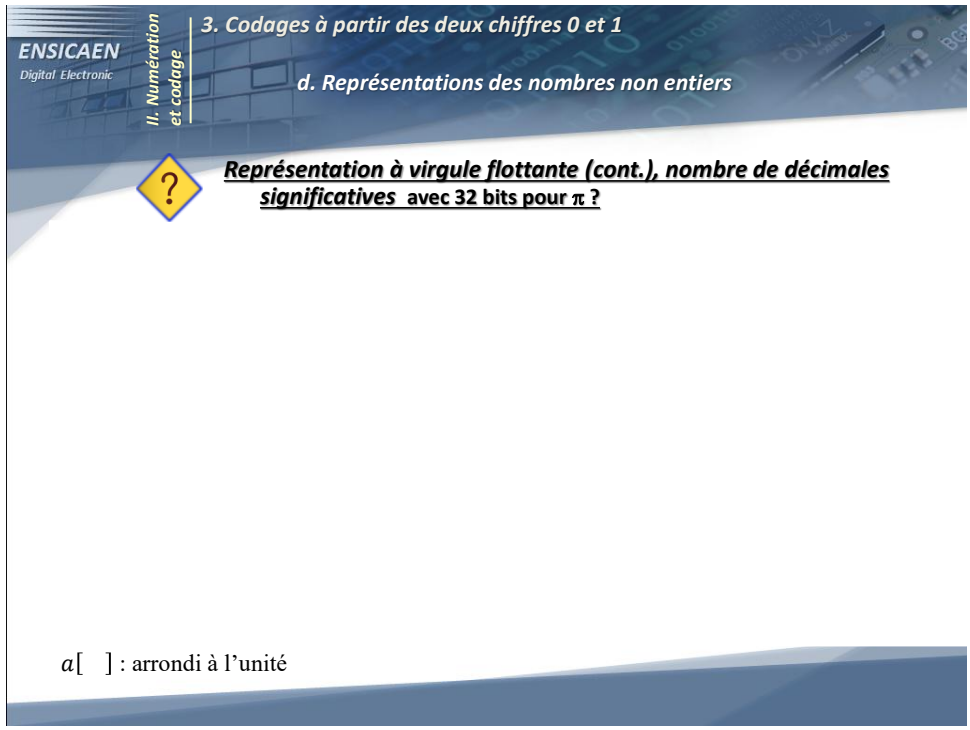
Norme IEEE 754 (cont.)

Exemples : sur 32 bits

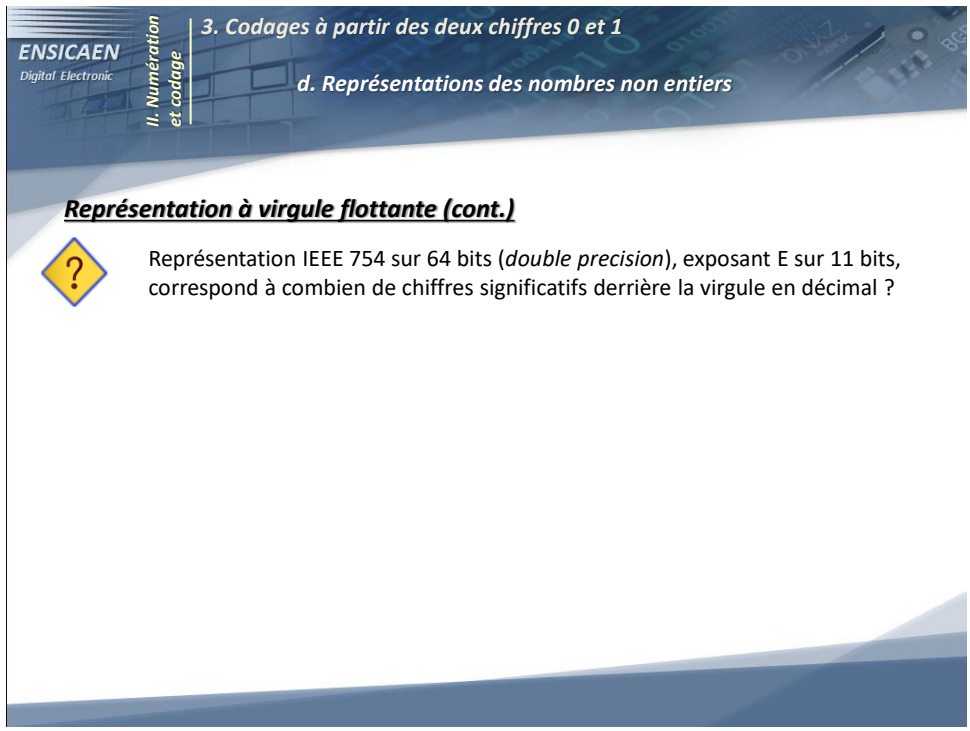
valeur	S	$E_7E_6E_5E_4E_3E_2E_1E_0$	$F_{22}F_{21}F_{20}....F_2F_1F_0$
0	0	0000 0000	000……000
-0	1	0000 0000	000……000
∞	0	1111 1111	000……000
-∞	1	1111 1111	000……000
NaN	0	1111 1111	00010….0
6,5	0	1000 0001	1010…..00
$7,347.10^{-39}$	0	0000 0000	1010…..00

41

41


ENSICAEN
Digital Electronic
II. Numération et codage
3. Codages à partir des deux chiffres 0 et 1
d. Représentations des nombres non entiers
Représentation à virgule flottante (cont.), nombre de décimales significatives avec 32 bits pour π ?
a[ ] : arrondi à l'unité

42


ENSICAEN
Digital Electronic
II. Numération et codage
3. Codages à partir des deux chiffres 0 et 1
d. Représentations des nombres non entiers
Représentation à virgule flottante (cont.)
Représentation IEEE 754 sur 64 bits (double precision), exposant E sur 11 bits, correspond à combien de chiffres significatifs derrière la virgule en décimal ?

43

Comparaison virgule fixe/ virgule flottante

Des critères de comparaisons sont la **dynamique de représentation**, le **bruit de quantification** et la **complexité des opérateurs arithmétiques**.

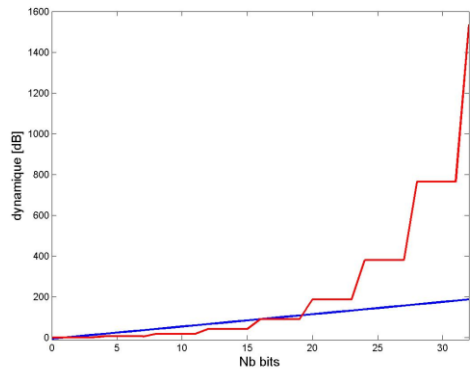
La dynamique se définie comme le rapport des valeurs maximum et minimum de la représentation.

$$dynamique = \left| \frac{valeur\max}{valeur\min} \right|$$

capacité à représenter des valeurs très petites (résolution) et très grandes (étendue)

On l'utilise souvent en dB par : $D_N(dB) = 20\log\left(\left|\frac{valeur\max}{valeur\min}\right|\right)$

Comparaison virgule fixe/ virgule flottante (cont.)



virgule fixe 16 bits : D_N(dB)=6.02.(16-1)=90.3 dB
virgule flottante 16 bits, E sur 4 bits : D_N(dB)=6.02.(2^4-1)=90.3 dB

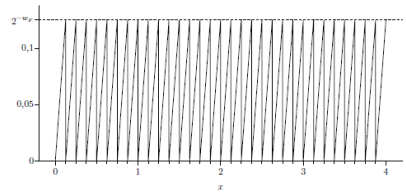
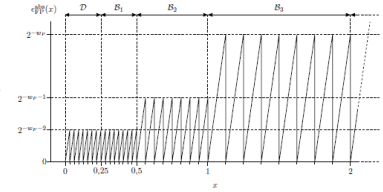
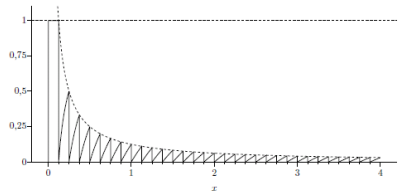
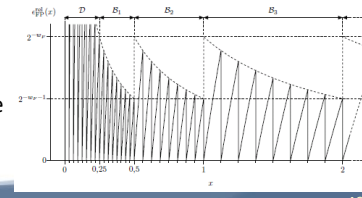
ENSICAEN
Digital Electronic
II. Numération et codage

3. Codages à partir des deux chiffres 0 et 1

d. Représentations des nombres non entiers

Comparaison virgule fixe/ virgule flottante (cont.)

Critère : bruit de quantification (erreur de représentation relative).

	virgule fixe	virgule flottante
erreur absolue (e)		
erreur relative (e/x)		

46

46

ENSICAEN
Digital Electronic
II. Numération et codage

3. Codages à partir des deux chiffres 0 et 1

e. Remarque déclaration de format de nombre en C

Rmq

Remarques : *déclaration de format de nombre en C*

Une déclaration de type de nombre sert à définir :

- *la taille (multiple de 8 bits).*
- *le domaine qui peut varier suivant la nature du type (signé, non signé).*

type	taille	domaine
unsigned char	8 bits	0 à 255
char	8 bits	-128 à 127
unsigned int (OS 32 bits)	32 bits	0 à 4 294 967 295
short int	16 bits	-32768 à 32767
int (OS 32 bits)	32 bits	-2 147 483 648 à 2 147 483 647
long	32 bits	-2 147 483 648 à 2 147 483 647
float	32 bits	$3{,}4.10^{-38}$ à $3{,}4.10^{38}$
double	64 bits	$1{,}7.10^{-308}$ à $1{,}7.10^{308}$

47

3. Codages à partir des deux chiffres 0 et 1

e. Remarque déclaration de format de nombre en C

Remarques : *déclaration de format de nombre en C*

```
uint_8 nb = 10001111;

itoa(nb, nbText, 10);
PutString(nbText);
```

→ 143

```
char nb = 10001111;

itoa(nb, nbText, 10);
PutString(nbText);
```

→ -113

Même contenu mémoire, interprétation différente

48

4. De l'importance du choix de codage

En fonction de la vulnérabilité de l'application : le code doit permettre la représentation des grandeurs manipulées sans débordement (ou avec signalisation du débordement) et avoir une résolution suffisante pour que le bruit de quantification (dit autrement l'arrondi de représentation) n'est pas d'impact sur l'application.

Le compromis est à trouver entre le nombre de bits utilisés pour la représentation et la dynamique associée et la taille/consommation/coût de l'unité de calcul associée.

3+2,19 =

49

49

4. De l'importance du choix de codage

Exemple d'effet d'arrondi :
système réel : incrément de 1/10 toutes 100 ms.

en théorie :

	temps [s]	valeur sur 8 bit	valeur sur 24 bits
	0	0	0
	0,1	0,09375	0,1
6h15 de décalage	0,2	0,1875	0,2
0,34s de décalage	...		...
100h	360000	337500	359999,66
	...		...
1 semaine	604800	567000	604799,42

en réalité :
1/10 ⇨ base 2 : 0.00011001100110011001…
sur 8 bits : 0.0001100 soit 0,09375
erreur de $6{,}25\times10^{-3}$ à chaque ajout
au bout de 100h : 6h15 de décalage
sur 24 bits :
erreur de $9{,}5367\times10^{-8}$ à chaque ajout
au bout de 100h : 0,34s de décalage
au bout d'une semaine : 0,576782 s

```
>>> 0.1+0.1 == 0.2
True
>>> 0.1+0.1+0.1 == 0.3
False
>>>
```

50

4. De l'importance du choix de codage

Appendix II

Effect of Extended Run Time on Patriot Operation

Hours	Seconds	Calculated Time (Seconds)	Inaccuracy (Seconds)	Approximate Shift In Range Gate (Meters)
0	0	0	0	0
1	3600	3599.9966	.0034	7
8	28800	28799.9725	.0025	55
20 (a)	72000	71999.9313	.0687	137
48	172800	172799.8352	.1648	330
72	259200	259199.7528	.2472	494
100 (b)	360000	359999.6667	.3433	687

a. Continuous operation exceeding about 20 hours--target outside range gate
b. Alpha Battery ran continuously for about 100 hours

Page 15 -- GAO/IMTEC-92-26 Patriot Missile Software Problem

erreur de distance = erreur de temps × vitesse

~2000 m/s

51

51

ENSICAEN
Digital Electronic
II. Numération et codage

4. De l'importance du choix de codage

Exemple de sous capacité de représentation :

- The internal SRI software exception was caused during execution of a data conversion from 64-bit floating point to 16-bit signed integer value. The floating point number which was converted had a value greater than what could be represented by a 16-bit signed integer. This resulted in an Operand Error. The data conversion instructions (in Ada code) were not protected from causing an Operand Error, although other conversions of comparable variables in the same place in the code were protected.
- The error occurred in a part of the software that only performs alignment of the strap-down inertial platform. This software module computes meaningful results only before lift-off. As soon as the launcher lifts off, this function serves no purpose.
- The alignment function is operative for 50 seconds after starting of the Flight Mode of the SRIs which occurs at H_0 - 3 seconds for Ariane 5. Consequently, when lift-off occurs, the function continues for approx. 40 seconds of flight. This time sequence is based on a requirement of Ariane 4 and is not required for Ariane 5.
- The Operand Error occurred due to an unexpected high value of an internal alignment function result called BH, Horizontal Bias, related to the horizontal velocity sensed by the platform. This value is calculated as an indicator for alignment precision over time.
- The value of BH was much higher than expected because the early part of the trajectory of Ariane 5 differs from that of Ariane 4 and results in considerably higher horizontal velocity values.

Ariane 501 Inquiry Board report
page 4

Ariane 501 Inquiry Board report

coût 500 millions !

1996

ARIANE 5

light 501 Failure

valeur plus grande que la représentation possible sur 16 bits

accélération Ariane 5 5 fois plus grande que Ariane 4

52

52

ENSICAEN
Digital Electronic
II. Numération et codage

4. De l'importance du choix de codage

Quand va-t-on avoir un problème avec le format du temps sur ordinateur ?

Représentation du temps sur les ordinateurs (heure POSIX ou heure UNIX) :

32bits entier signé pour représenter le nombre de secondes depuis le 1er janvier 1970 à minuit.

Maximum :

54

ENSICAEN
Digital Electronic

II. Numération et codage

A l'issue de cette deuxième partie (1/3)

- Comment sont représentées d'un point de vue électrique les valeurs numériques dans les composants numériques (circuits logiques, microprocesseurs, microcontrôleurs) ?
- Dans quels cas utilise-t-on la base 10 ?
- Pourquoi utilise-t-on la base 16 ?
- Qu'est-ce qu'un code pondéré ?

55

55

ENSICAEN
Digital Electronic

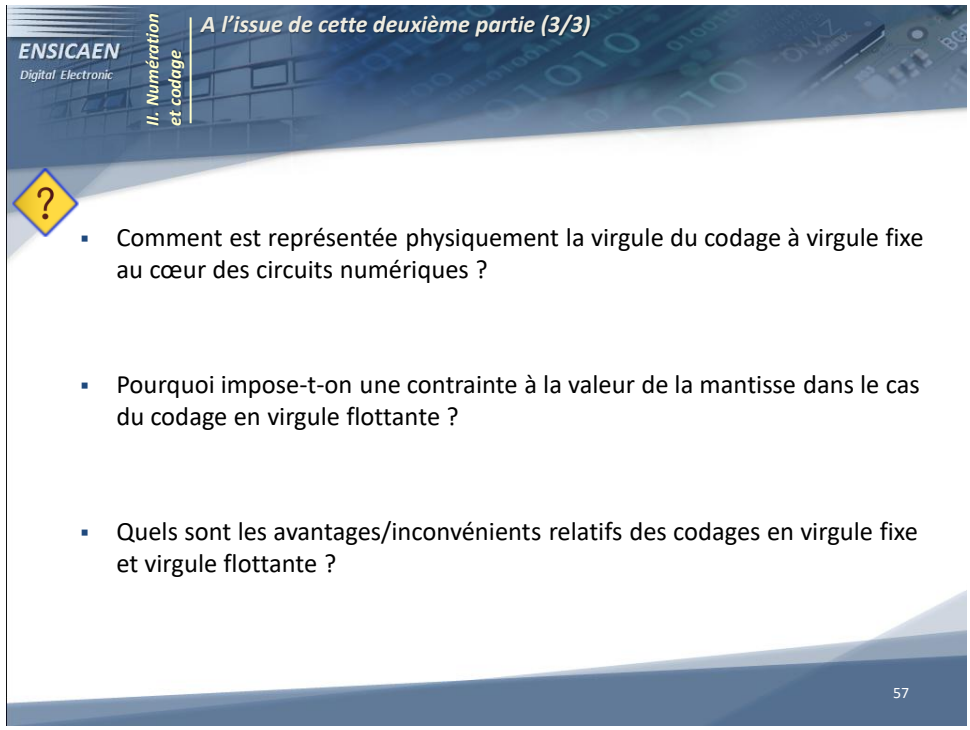
II. Numération et codage

A l'issue de cette deuxième partie (2/3)

- Quelles caractéristiques doit présenter un codage utilisé pour le calcul ?
- Quel codage est utilisé en pratique dans les circuits numériques pour représenter les nombres positifs et négatifs ?
- Comment représente-t-on un nombre positif en complément à 2 ?
- Comment représente-t-on un nombre négatif en complément à 2 ?

56

56


ENSICAEN
Digital Electronic
II. Numération et codage
A l'issue de cette deuxième partie (3/3)
?
Comment est représentée physiquement la virgule du codage à virgule fixe au cœur des circuits numériques ?
Pourquoi impose-t-on une contrainte à la valeur de la mantisse dans le cas du codage en virgule flottante ?
Quels sont les avantages/inconvénients relatifs des codages en virgule fixe et virgule flottante ?
57

57

ENSICAEN
Digital Electronic

III. Logique combinatoire

Algèbre de Boole et fonctions Booléennes, implémentation

M. Denoual

1

1

Back to basics

développée de longue date

de λόγος / lógos signifiant
raison, langage et raisonnement

- Conceptuellement :
  - En philosophie (le vrai, le faux) : étude des règles formelles que doit respecter toute argumentation correcte. III^e siècle av. J.-C.
  - En mathématique : algèbre de Boole. XIX^e siècle.
- En pratique :
  - Hydraulique
  - Mécanique

Steve Mould, Water computing

Pascaline, 1642

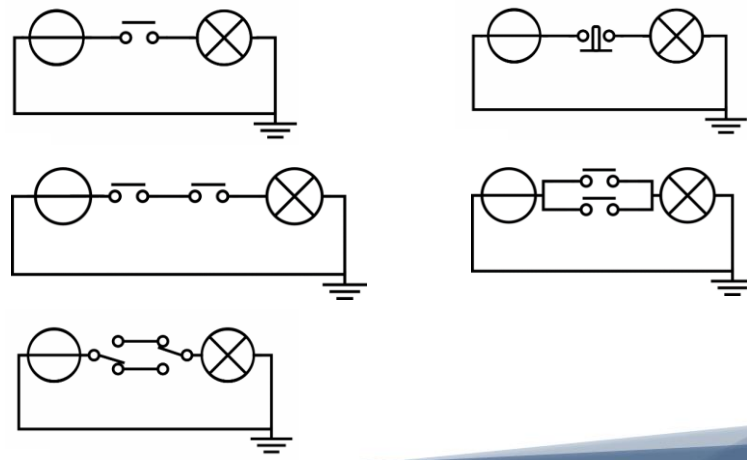
  - Électrique (électricité/relais-interrupteurs diodes → transistors)

2

2

ENSICAEN
Digital Electronic
III. Logique combinatoire
Back to basics
logique d'interrupteur

- Électricité : logique d'interrupteur



3

3

ENSICAEN
Digital Electronic
III. Logique combinatoire
Algèbre de Boole : Opérations de base

- Algèbre de Boole définie sur E_2 constitué des éléments {0,1}
- Trois opérations de base :

complémentation		union + max OU			intersection · min ET		
a	$\overline{a}$	a	b	s	a	b	s
0	1	0	0	0	0	0	0
1	0	0	1	1	0	1	0
		1	0	1	1	0	0
		1	1	1	1	1	1

- 0 : élément neutre de l'union, élément minimum.
- 1 : élément neutre de l'intersection, élément maximum.

4

4

ENSICAEN
Digital Electronic
III. Logique combinatoire

Algèbre de Boole : Propriétés des opérations de base

$\forall a, b, c \in E_2$

- complémentation : $a.\bar{a} = 0$; $a + \bar{a} = 1$
- commutativité : $a.b = b.a$; $a + b = b + a$
- associativité : $(a.b).c = a.(b.c)$; $(a + b) + c = a + (b + c)$
- distributivité : $a.(b + c) = a.b + a.c$; $a + (b.c) = (a + b).(a + c)$

5

5

ENSICAEN
Digital Electronic
III. Logique combinatoire

Algèbre de Boole : Théorèmes associés à l'algèbre de Boole

$\forall a, b, c \in E_2$

- idempotence : $a.a = a$; $a + a = a$
- absorption : $a + a.b = a$; $a.(a + b) = a$
- involution : $\bar{\bar{a}} = a$
- Théorème de Morgan : $\overline{a + b} = \bar{a}.\bar{b}$; $\overline{a.b} = \bar{a} + \bar{b}$
- Théorème de Shannon : $a + b = a.b + a.\bar{b} + \bar{a}.b$
- Consensus de 1ière espèce : $a + \bar{a}.b = a + b$; $a.(\bar{a} + b) = a.b$
- Consensus de 2nd espèce : $a.b + \bar{b}.c = a.b + \bar{b}.c + a.c$; $(a + b).(\bar{b} + c) = (a + b).(\bar{b} + c).(a + c)$
  b variable biforme

Termes de Consensus

6

6

ENSICAEN
Digital Electronic
III. Logique combinatoire

Algèbre de Boole : Méthode de réduction des expressions algébriques

à voir par vous même

- repérer les variables biformes,
- introduire le terme de Consensus de 2nd espèce s'il permet de réduire l'expression par absorption,
- absorption,
- retirer le terme de consensus.

Exemple :

$F = f\bar{e}b + hgf\bar{e}dca + \mathrm{hg}\bar{b}$

ajout terme Consensus

$F = f\bar{e}b + hgf\bar{e}dca + \mathrm{hg}\bar{b} + \mathrm{hgf}\bar{e}$

Absorption
retrait Consensus

$F = f\bar{e}b + \mathrm{hg}\bar{b}$

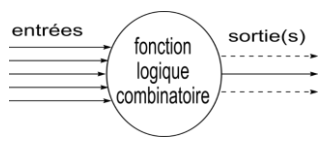
7

7

ENSICAEN
Digital Electronic
III. Logique combinatoire

Algèbre de Boole : Fonctions booléennes

$E_2 x E_2 x ... x E_2$ dans E_2.



entrées		sortie(s)
a	b	s
0	0	0
0	1	1
1	0	1
1	1	0

Table de vérité

Objectif :
Exprimer la ou les sorties en fonction des entrées (puis y associer des portes logiques – support matériel : synthèse logique)

Deux façons d'exprimer une fonction booléenne :

- la forme **somme de produit** ou disjonctive normale ΣΠ
- la forme **produit de somme** ou conjonctive normale ΠΣ

Idée : avoir l'expression la plus compacte possible

Moins de support physique, moins de consommation, plus rapide

8

8

Algèbre de Boole : Forme somme de produit ΣΠ

- Première forme canonique
  - On associe une variable binaire m_i à chaque entrée de la table
    - *minterm* , terme produit

m_i	a	b	s
m_0	0	0	s_0
m_1	0	1	s_1
m_2	1	0	s_2
m_3	1	1	s_3

m_0 est associé à la ligne 0, actif/vrai (à 1) pour la ligne 0

$m_0 = 1$ si $a = 0$ et b=0 c'est-à-dire si $\bar{a} = 1$ et $\bar{b} = 1$

$m_0 = \bar{a}.\bar{b}$

$m_0 = \bar{a}.\bar{b}$; $m_1 = \bar{a}.b$; $m_2 = a.\bar{b}$; $m_3 = a.b$

$$s = s_0 m_0 + s_1 m_1 + s_2 m_2 + s_3 m_3$$

9

9

Algèbre de Boole : Forme somme de produit ΣΠ (cont.)

- Simplification de l'expression
  - minterm pour lesquels la fonction vaut 1
  - Dans le cas de la fonction XOR

m_i	a	b	s
m_0	0	0	0 (s_0)
m_1	0	1	1 (s_1)
m_2	1	0	1 (s_2)
m_3	1	1	0 (s_3)

$$s = m_1 + m_2$$

$$s = \bar{a}.b + a.\bar{b}$$

Rmq **Remarque :** réalisation simple avec des portes NAND

10

10

ENSICAEN
Digital Electronic
III. Logique combinatoire

Algèbre de Boole : Forme produit de somme ΠΣ

à voir par vous même

- Seconde forme canonique
  - On associe une variable binaire M_i à chaque entrée de la table
    - *maxterm*, terme somme

M_i	a	b	s
M_0	0	0	0 (s_0)
M_1	0	1	1 (s_1)
M_2	1	0	1 (s_2)
M_3	1	1	0 (s_3)

- Exemple pour la fonction XOR

a	b	M_0	M_1	M_2	M_3
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

Maxterm constitué de l'union (OU) de toutes les variables d'entrée, non complémentées si leur valeur est 0, complémentées si leur valeur est 1.

$$M_0 = a + b\,;M_1 = a + \bar{b}\,;M_2 = \bar{a} + b\,;M_3 = \bar{a} + \bar{b}$$

$$s = (s_0 + M_0).(s_1 + M_1).(s_2 + M_2).(s_3 + M_3)$$

11

11

ENSICAEN
Digital Electronic
III. Logique combinatoire

Algèbre de Boole : Forme produit de somme ΠΣ (cont.)

à voir par vous même

- Simplification de l'expression
  - On conserve les maxterm pour lesquels la fonction vaut 0

a	b	M_0	M_1	M_2	M_3	M_0M_3	s
0	0	0	1	1	1	0	0
0	1	1	0	1	1	1	1
1	0	1	1	0	1	1	1
1	1	1	1	1	0	0	0

$s = 0$ si $M_0 = 0$ ou si $M_3 = 0$ c'est-à-dire si $M_0.M_3 = 0$

$s = M_0.M_3$ $\qquad$ $s = (a + b).(\bar{a} + \bar{b})$

Rmq **Remarque :** réalisation simple avec des portes NOR

12

12

ENSICAEN
Digital Electronic
III. Logique combinatoire
Fonctions logiques
Portes logiques

Aux fonctions logiques booléennes élémentaires sont associées des portes logiques.

Ces portes logiques ont une existence matérielle qui permet de réaliser concrètement les fonctions logiques.

Elles peuvent se présenter sous différentes formes:
- discrètes,
- intégrées,
- assemblées dans des blocs élémentaires,
- sous forme de matrice connectables ,
- …

Idée : passer d'une fonction sur papier ou écran de PC à un support physique

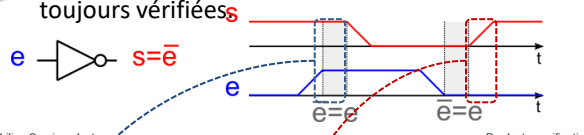
fonction		symbole CEI 60617 EN 60617:1999	(ancien symbole)	table de vérité
NON, NOT, inversion	-			a s: 0 1 / 1 0
ET, AND, intersection, min	.			a b s: 0 0 0 / 0 1 0 / 1 0 0 / 1 1 1
OU, OR, union, max	+			a b s: 0 0 0 / 0 1 1 / 1 0 1 / 1 1 1
NAND, non-ET	/			a b s: 0 0 1 / 0 1 1 / 1 0 1 / 1 1 0
NOR, non-OU	↓			a b s: 0 0 1 / 0 1 0 / 1 0 0 / 1 1 0
XOR, ou-exclusif	⊕			a b s: 0 0 0 / 0 1 1 / 1 0 1 / 1 1 0

13

13

ENSICAEN
Digital Electronic
III. Logique combinatoire
Fonctions logiques
Portes logiques

Rmq **Remarque 1** : en pratique, les équations de l'algèbre de Boole ne sont pas toujours vérifiées



Philips Semiconductors — Product specification

Quadruple exclusive-OR gate — HEF4030B gates

	V_{DD} V	SYMBOL	TYP.	MAX.		TYPICAL EXTRAPOLATION FORMULA
Propagation delays $I_n \rightarrow O_n$ HIGH to LOW	5	t_{PHL}	85	175	ns	57 ns + (0,55 ns/pF) C_L
	10		35	75	ns	24 ns + (0,23 ns/pF) C_L
	15		30	55	ns	22 ns + (0,16 ns/pF) C_L
LOW to HIGH	5	t_{PLH}	75	150	ns	47 ns + (0,55 ns/pF) C_L
	10		30	65	ns	19 ns + (0,23 ns/pF) C_L
	15		25	50	ns	17 ns + (0,16 ns/pF) C_L
Output transition times HIGH to LOW	5	t_{THL}	60	120	ns	10 ns + (1,0 ns/pF) C_L
	10		30	60	ns	9 ns + (0,42 ns/pF) C_L
	15		20	40	ns	6 ns + (0,28 ns/pF) C_L
LOW to HIGH	5	t_{TLH}	60	120	ns	10 ns + (1,0 ns/pF) C_L
	10		30	60	ns	9 ns + (0,42 ns/pF) C_L
	15		20	40	ns	6 ns + (0,28 ns/pF) C_L

Idée : différence entre idéal (fonction logique) et implémentation (réalisation physique)

important !

Physiquement :
Temps de propagation
Temps d'établissement (montée, descente)
Limites en courant (sortance)
Consommation

14

14

ENSICAEN
Digital Electronic
III. Logique combinatoire
Fonctions logiques
Portes logiques
Synthèse combinatoire

Synthèse combinatoire

fonction logique à réaliser sous forme de table ou d'équation

choix d'une cible technologique

réalisation matérielle de la fonction

- logique anarchique
- opérateurs élémentaires
- logique structurée (mémoire ROM, PAL, CPLD, FPGA)

16

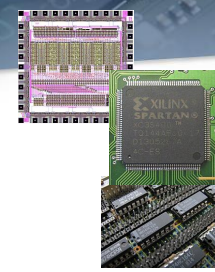
16

ENSICAEN
Digital Electronic
III. Logique combinatoire
Fonctions logiques
Portes logiques
Synthèse combinatoire

Synthèse combinatoire

fonction logique à réaliser sous forme de table ou d'équation

choix d'une cible technologique



Note

le choix de la cible technologique dépend
- de la complexité de la fonction à réaliser
- du coût
- de la vitesse de traitement nécessaire
- du savoir-faire du concepteur
- …

réalisation matérielle de la fonction

- logique anarchique
- opérateurs élémentaires
- logique structurée (mémoire ROM, PAL, CPLD, FPGA)

17

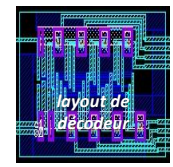
17

ENSICAEN
Digital Electronic
III. Logique combinatoire

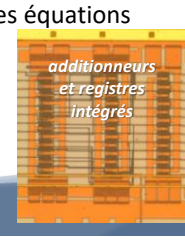
Synthèse combinatoire – 1- Logique anarchique

- à base de porte logique AND, OR, NAND, NOR, XOR, INV.
  - méthode historique
  - adaptée à de très faibles complexités dans le cas de composants discrets
  - utilisation de composants discrets en voie d'obsolescence

  - base de pour les circuits intégrés full-custom
    - (*ASIC: Application Specific Integrated Circuit*)
  - création de bloc élémentaire pour des structures plus complexes
    - cas des opérateur arithmétiques



- première étape de la synthèse, simplification des équations
  - algèbre de Boole
  - table de Karnaugh
  - logiciel de synthèse



18

18

ENSICAEN
Digital Electronic
III. Logique combinatoire

Synthèse combinatoire - 1- Logique anarchique - Table de Karnaugh

Écrivez les équations Booléennes d'une fonction active pour les nombres premiers entre 0 et 7

23

23

ENSICAEN
Digital Electronic
III. Logique combinatoire

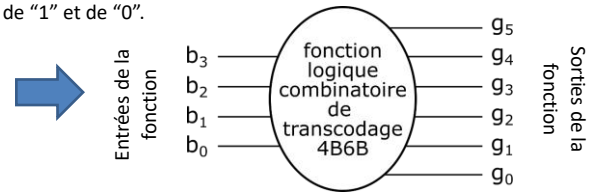
Synthèse combinatoire - 1- Logique anarchique - Table de Karnaugh

Exemple du transcodeur 4B6B

Le transcodage 4B6B étend un code de 4 bits en un code de 6 bits contenant le même nombre de "1" et de "0".

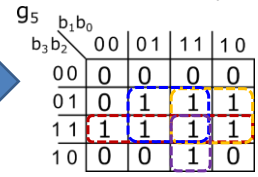
	Code binaire 4B	Code 6B
	$b_3\ b_2\ b_1\ b_0$	$g_5\ g_4\ g_3\ g_2\ g_1\ g_0$
0	0 0 0 0	0 0 1 1 1 0
1	0 0 0 1	0 0 1 1 0 1
2	0 0 1 0	0 1 0 0 1 1
3	0 0 1 1	0 1 0 1 1 0
4	0 1 0 0	0 1 0 1 0 1
5	0 1 0 1	1 0 0 0 1 1
6	0 1 1 0	1 0 0 1 1 0
7	0 1 1 1	1 0 0 1 0 1
8	1 0 0 0	0 1 1 0 0 1
9	1 0 0 1	0 1 1 0 1 0
A	1 0 1 0	0 1 1 1 0 0
B	1 0 1 1	1 1 0 0 0 1
C	1 1 0 0	1 1 0 0 1 0
D	1 1 0 1	1 0 1 0 0 1
E	1 1 1 0	1 0 1 0 1 0
F	1 1 1 1	1 0 1 1 0 0

Table de vérité de la fonction logique de transcodage 4B6B



Fonction logique combinatoire à 4 entrées et 6 sorties.

Exemple de résolution pour la sortie g_5 Il faut faire de même pour les 5 autres sorties (g_0 à g_4).



$$g_5 = b_2b_1 + b_2b_0 + b_3b_2 + b_3b_1b_0$$

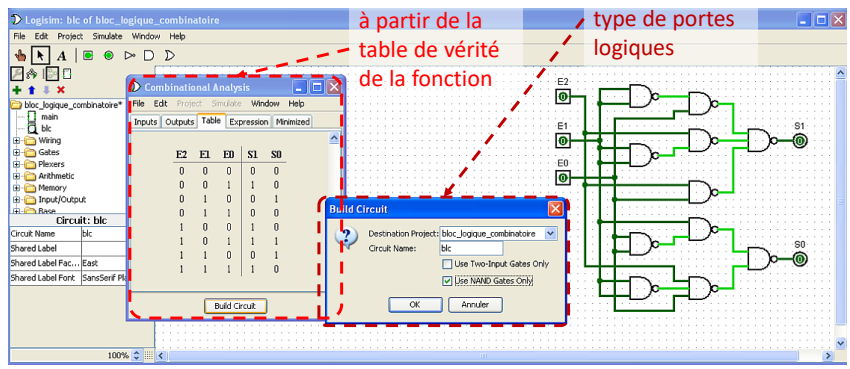
24

24

ENSICAEN
Digital Electronic
III. Logique combinatoire

Synthèse combinatoire - 1- Logique anarchique – Logiciel de synthèse logique

- algorithme de réduction algébrique.



Exemple avec le logiciel gratuit Logisim

Logisim: Download (cburch.com)

25

25

ENSICAEN Digital Electronic — III. Logique combinatoire — Synthèse combinatoire - 1- Logique anarchique – Logiciel de synthèse logique

Exemple de l'algorithme de Quine-Mac Cluskey (1956)

utilise la 1ère forme canonique (ΣΠ). Les combinaisons sont rassemblées en groupes en fonction du nombre de 1 des minterms pour lesquels la fonction est vraie. Des simplifications sont faites entre les minterms adjacents de groupes consécutifs.

Sur un exemple :

d	c	b	a	f
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

rassemblement des minterms

N° minterm

grp	N°	d	c	b	a
0	0	0	0	0	0
1	1	0	0	0	1
	2	0	1	0	0
2	3	0	1	0	1
	4	1	0	0	1
	5	1	0	1	0
3	6	0	1	1	1
	7	1	1	0	1
4	8	1	1	1	1

26

26

ENSICAEN Digital Electronic — III. Logique combinatoire — Synthèse combinatoire - Logique anarchique – Logiciel de synthèse logique

grp	N°	d	c	b	a
0	0	0	0	0	0
1	1	0	0	0	1
	2	0	1	0	0
2	3	0	1	0	1
	4	1	0	0	1
	5	1	0	1	0
3	6	0	1	1	1
	7	1	1	0	1
4	8	1	1	1	1

simplifications

grp	N°	d	c	b	a
0-1	0-1	0	0	0	X
	0-2	0	X	0	0
1-2	1-4	X	0	0	1
	2-3	0	1	0	X
2-3	3-6	0	1	X	1
	3-7	X	1	0	1
	4-7	1	X	0	1
3-4	6-8	X	1	1	1
	7-8	1	1	X	1

le N°5 n'est pas représenté

comparaison du minterm du groupe 0 avec les minterms du groupe 1

0-1	0	0	0	X
0-2	0	X	0	0

comparaison des minterms du groupe 1 avec les minterms du groupe 2

1-4	X	0	0	1
2-3	0	1	0	X

comparaison des minterms du groupe 2 avec les minterms du groupe 3

3-6	0	1	X	1
3-7	X	1	0	1
4-7	1	X	0	1

comparaison des minterms du groupe 3 avec le minterm du groupe 4

6-8	X	1	1	1
7-8	1	1	0	1

27

27

ENSICAEN
Digital Electronic
III. Logique combinatoire

Synthèse combinatoire - Logique anarchique – Logiciel de synthèse logique

grp	N°	d	c	b	a
0-1	0-1	0	0	0	X
	0-2	0	X	0	0
1-2	1-4	X	0	0	1
	2-3	0	1	0	X
2-3	3-6	0	1	X	1
	3-7	X	1	0	1
	4-7	1	X	0	1
3-4	6-8	X	1	1	1
	7-8	1	1	X	1

le N°5 n'est pas représenté

simplifications

grp	N°	d	c	b	a
0-1 ; 1-2	0-1 ; 2-3	0	X	0	X
1-2 ; 2-3	1-4 ; 3-7	X	X	0	1
2-3 ; 3-4	3-6 ; 7-8	X	1	X	1
	3-7 ; 6-8	X	1	X	1

le N°5 n'est pas représenté

comparaison du minterm du groupe 0-1
avec les minterms du groupe 1-2

0-1 ; 2-3 0 X 0 X

comparaison des minterms du groupe 1-2
avec les minterms du groupe 2-3

1-4 ; 3-7 X X 0 1

comparaison des minterms du groupe 2-3
avec les minterms du groupe 3-4

3-6 ; 7-8 X 1 X 1

3-7 ; 6-8 X 1 X 1

28

28

ENSICAEN
Digital Electronic
III. Logique combinatoire

Synthèse combinatoire - 1- Logique anarchique – Logiciel de synthèse logique

grp	N°	d	c	b	a
0-1 ; 1-2	0-1 ; 2-3	0	X	0	X
1-2 ; 2-3	1-4 ; 3-7	X	X	0	1
2-3 ; 3-4	3-6 ; 7-8	X	1	X	1
	3-7 ; 6-8	X	1	X	1

le N°5 n'est pas représenté

1er réarrangement

grp	N°	d	c	b	a
0	0	0	0	0	0
1	1	0	0	0	1
	2	0	1	0	0
2	3	0	1	0	1
	4	1	0	0	1
	5	1	0	1	0
3	6	0	1	1	1
	7	1	1	0	1
4	8	1	1	1	1

Finalement, la fonction est représentée par : (5), (0-1 ; 2-3), (1-4 ; 3-7), (3-6 ; 7-8)

ou encore : $d\bar{c}b\bar{a} + \overline{d}\overline{b} + \bar{b}a + ca$

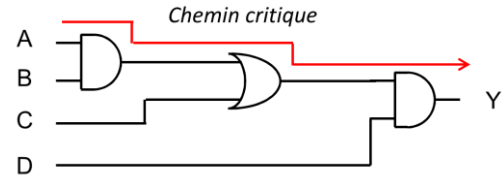
29

29

Chemin critique

– Le parcours qui va générer le délai de propagation le plus long entre les entrées et la sortie du circuit. C'est a priori le chemin qui traverse le plus de portes logiques.

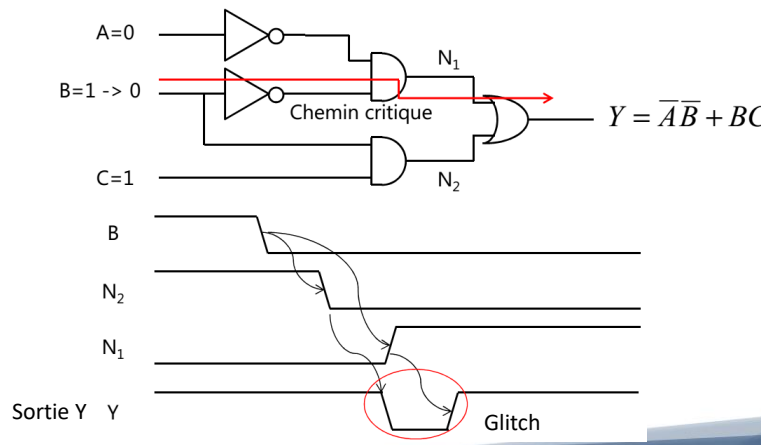
porte	Délai (ns)
NOT	30
AND	60
OR	70



Rmq Le chemin critique détermine le temps de calcul des opérateurs arithmétiques pour le calcul.

Rmq Le chemin critique limite la fréquence maximum de fonctionnement des systèmes synchrones.

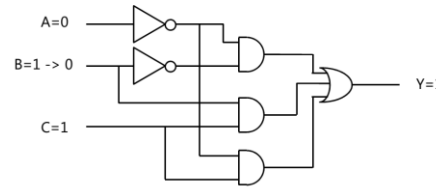
Glitch : variations transitoires des niveaux logiques avant d'atteindre un état



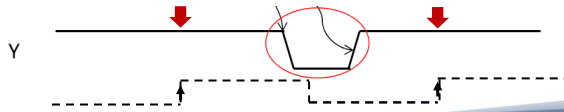
Glitch : solutions

C \ BA	00	01	11	10
0	1	0	0	0
1	1	0	1	1

1. Les éliminer en modifiant le circuit (par exemple termes redondants de Consensus)



2. Ne pas s'en préoccuper en utilisant une **conception synchrone** avec un cadencement d'horloge adapté (on ne prend en compte le signal que périodiquement, une fois stabilisé).

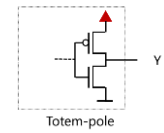


Sortie d'un circuit logique

Types de sortie (étage de sortie): la sortie d'un circuit logique peut être fournie soit par 2 transistors complémentaires « totem-pole » ou par un seul transistor « collecteur ouvert ».

La sortie totem-pole est pratique pour cascader des circuits de même technologie, même alimentation.

La sortie collecteur ouvert est pratique pour changer de niveau de tension ou la réalisation d'un ET câblé (bus I2C).

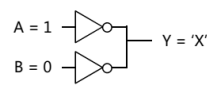


Totem-pole

Collecteur ouvert

Sortance : capacité d'une porte logique à servir de source à d'autres portes logiques. Courant de sortie > somme de courant d'entrée des portes en aval. En CMOS limite de vitesse de fonctionnement.

Court-circuit (état indéterminé)



Opérateurs élémentaires

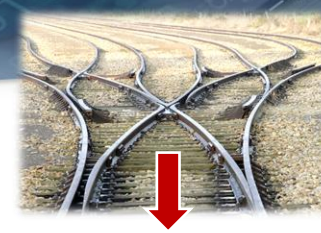
- multiplexeurs,
- démultiplexeur,
- décodeur,
- encodeur

Rmq Pédagogique, en pratique plus utilisé

Rmq opérateurs élémentaires eux-mêmes constitués de portes logiques ET, NON, OU…

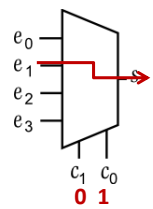
Rmq comme la logique anarchique avec des portes ET, OU, NON, … discrètes, ce type d'implémentation est en voie d'obsolescence et réservé à des fonctions logiques simples

Idée : opérateurs ne sont plus utilisés pour la synthèse logique mais sont des éléments de base des systèmes numériques



Multiplexeur

- **fonction d'aiguillage**
- exemple d'un multiplexeur 4 vers1

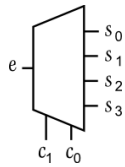


c_1	c_0	s
0	0	e_0
0	1	e_1
1	0	e_2
1	1	e_3

$$s = e_0.\overline{c_1}.\overline{c_0} + e_1.\overline{c_1}.c_0 + e_2.c_1.\overline{c_0} + e_3.c_1.c_0$$

Démultiplexeur

- opérateur dual



$$s_0 = \overline{c_1}.\overline{c_0}.e$$
$$s_1 = \overline{c_1}.c_0.e$$
$$s_2 = c_1.\overline{c_0}.e$$
$$s_3 = c_1.c_0.e$$

c_1	c_0	s_0	s_1	s_2	s_3
0	0	e	0	0	0
0	1	0	e	0	0
1	0	0	0	e	0
1	1	0	0	0	e

Décodeur

- active une sortie parmi 2^N grâce à N signaux de commande.
- **utilisé pour « adresser »**.
- équivalent du démultiplexeur avec une entrée de donnée à « 1 »

- exemple de décodeur 4 sorties

$$s_0 = \overline{c_1}.\overline{c_0}$$
$$s_1 = \overline{c_1}.c_0$$
$$s_2 = c_1.\overline{c_0}$$
$$s_3 = c_1.c_0$$

c_1	c_0	s_0	s_1	s_2	s_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Rmq Dans certains décodeurs discrets, la sortie active peut être au niveau bas et les inactives au niveau haut.

Logique structurée

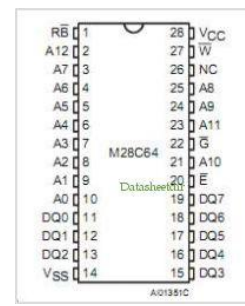
- structures universelles prédéfinies sous forme de matrice
- forte densité d'intégration → VLSI
- structure dépend du composant :
  - ROM, espace mémoire et système d'adressage → Note : *structure la plus universelle*
  - PAL, matrices de ET et de OU pouvant être interconnectées
  - CPLD, macrocellules logiques interconnectables
  - FPGA

Note : *toutes les fonctions logiques ne sont pas réalisables, mais il y de la marge*

- mémoire Mxn : sauvegarde de M mots de n bits.
- Exemple :

MC28C64C : 64 kbit ; 8kx8bits, parallel EEPROM

nombre de bits d'adresse ?



Idée : prétexte pour parler des mémoires, n'est plus utilisé en pratique pour la synthèse logique, mais reste un élément fondamental de l'électronique numérique

ENSICAEN
Digital Electronic
III. Logique combinatoire

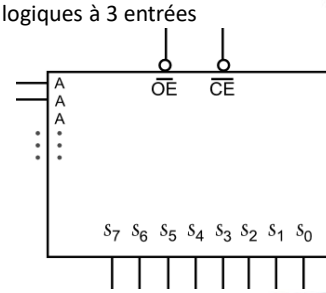
Synthèse combinatoire - 3- Logique structurée - Structure ROM (cont.)

- La taille du mot mémorisé permet de définir autant de fonctions logiques que son nombre de bits.
- Le nombre de bits d'adresse correspond au nombre de variables des fonctions logiques.

Remarque :
plus petites mémoires
2kx8 : 16k RAM 1,6€ (2021)
1k EEPROM : 0.31 € (Mouser 2025)
2k EEPROM : 0,23 € 24C02 (Mouser 2025)

- Exemple : implémentation de 5 fonctions logiques à 3 entrées

c	b	a	F_4	F_3	F_2	F_1	F_0
0	0	0	0	1	1	0	0
0	0	1	1	1	0	1	0
0	1	0	1	0	0	0	1
0	1	1	0	0	1	0	0
1	0	0	0	1	0	1	0
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	1
1	1	1	0	0	1	1	1



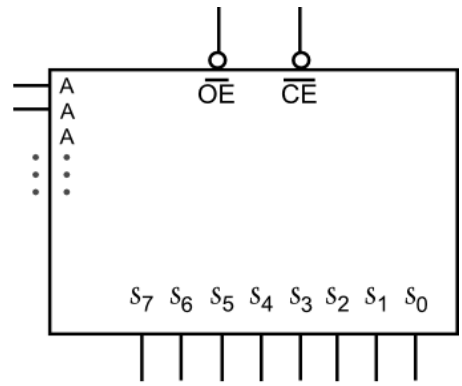
43

43

ENSICAEN
Digital Electronic
III. Logique combinatoire

Synthèse combinatoire - 3- Logique structurée - Structure ROM (cont.)

c	b	a	F_4	F_3	F_2	F_1	F_0
0	0	0	0	1	1	0	0
0	0	1	1	1	0	1	0
0	1	0	1	0	0	0	1
0	1	1	0	0	1	0	0
1	0	0	0	1	0	1	0
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	1
1	1	1	0	0	1	1	1



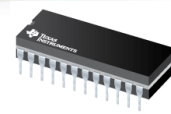
44

44

ENSICAEN
Digital Electronic
III. Logique combinatoire

Synthèse combinatoire - 3- Logique structurée - Structure PAL

historique (années 80)



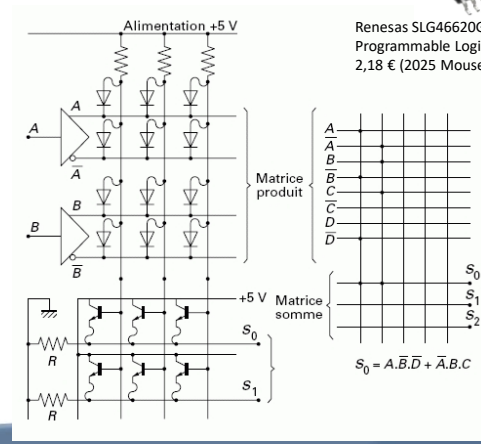
TICPAL22V10Z-25C 17,26 $
Texas Instrument 2022

Programmable Logic Array

- constituée de deux sous-ensembles :
  - une matrice de ET,
  - une matrice de OU.

PAL 16V8
- 16 : nb variables
- 8 : nb sorties

fabricants : Lattice Semiconductor, AMD (*Advanced Micro Devices*), Texas Instrument



Renesas SLG46620G Simple Programmable Logic Device (SPLD), 2,18 € (2025 Mouser)

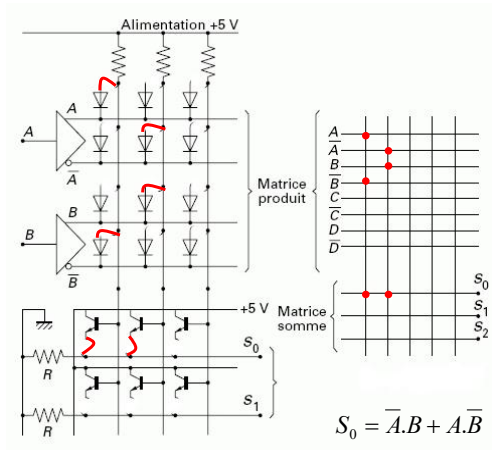
45

45

ENSICAEN
Digital Electronic
III. Logique combinatoire

Synthèse combinatoire - 3- Logique structurée - Structure PAL (cont.)

- implémentation de la fonction XOR



46

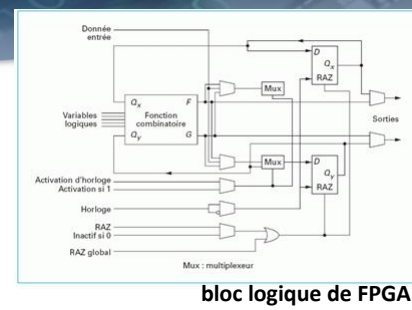
46

ENSICAEN
Digital Electronic
III. Logique combinatoire

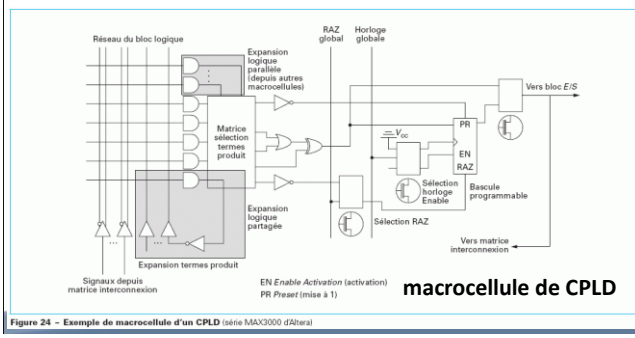
Synthèse combinatoire - 3- Logique structurée – Structures CPLD, FPGA

CPLD : Complex Programmable Logic Device
FPGA : Field Programmable Gate Array

- réseaux logiques programmables complexes formés de blocs logiques disposés sous forme matricielle



bloc logique de FPGA
Look-Up Table (LUT)



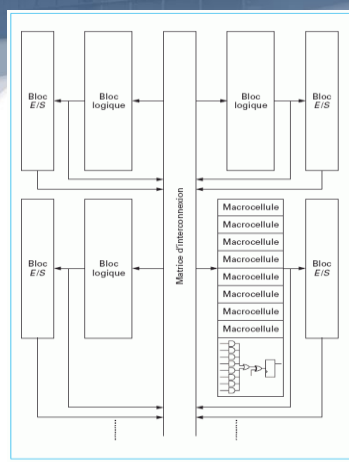
macrocellule de CPLD

Figure 24 – Exemple de macrocellule d'un CPLD (série MAX3000 d'Altera)

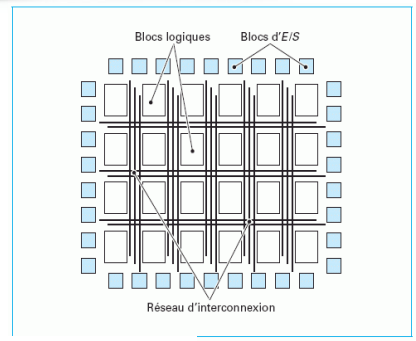
Idée : composants de l'électronique numérique programmée actuelle

ENSICAEN
Digital Electronic
III. Logique combinatoire

Synthèse combinatoire - 3- Logique structurée – Structures CPLD, FPGA (cont.)



structure CPLD



structure FPGA

FPGA plus interconnecté ⇨ plus flexible
⇨ plus difficile à implémenter
⇨ délais plus difficiles à estimer
FPGA plus de ressources intégrées

Note implémentation avec un logiciel et un langage de programmation

ENSICAEN
Digital Electronic
III. Logique combinatoire

Synthèse combinatoire - 3- Logique structurée – Structures CPLD, FPGA (cont.)

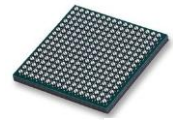
Fabricant FPGA : Xilinx, Altera, Lattice

AMD INTEL

Fabricant CPLD : Altera, Lattice, Atmel

Altera 5M40ZE64
32 macrocellules
3,44 € (Farnell 2022)
1,45 € (Mouser 2025)

Lattice ICE40 4,26 €
Radiospare 2022
1280 blocs logiques,
16kbits RAM mémoire



Artix 7 Xilinx 215360
macrocellules,
676 broches
433,86 € (Farnell 2022)
Artix 7 AMD
446,08 € (Mouser 2025)

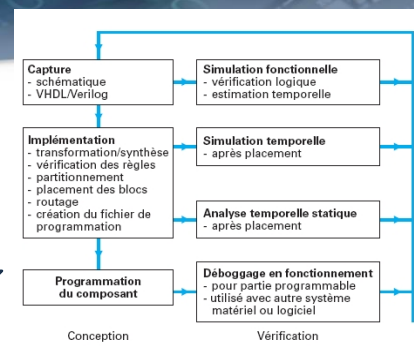
49

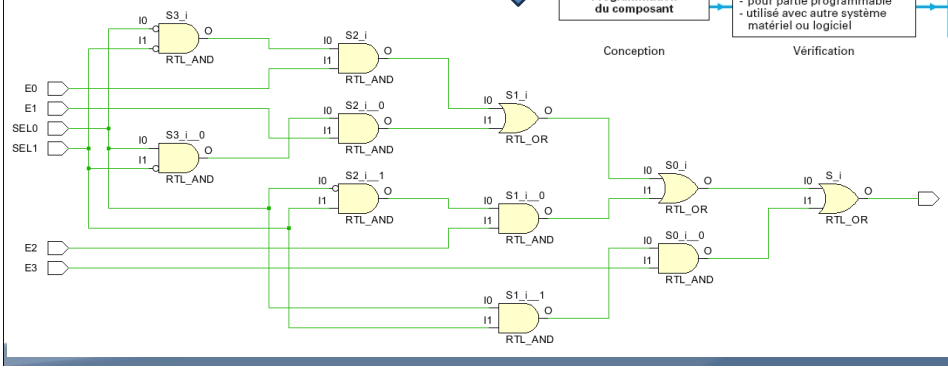
ENSICAEN
Digital Electronic
III. Logique combinatoire

Synthèse combinatoire - 3- Logique structurée – Structures CPLD, FPGA (cont.)

- VHDL, verilog

```
architecture Flow of MUX4 is
begin
    S <= ((not SEL0) and (not SEL1) and E0) or
         (SEL0 and (not SEL1) and E1) or
         ((not SEL0) and SEL1 and E2) or
         (SEL0 and SEL1 and E3);
end Flow;
```

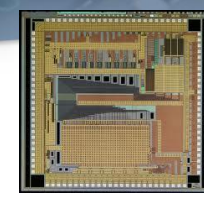




50

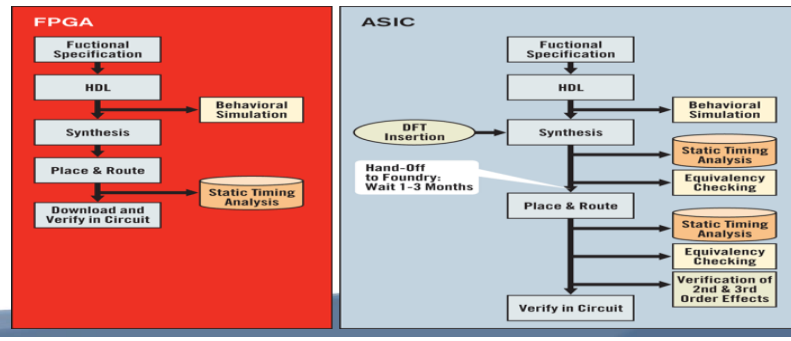
ENSICAEN
Digital Electronic
III. Logique combinatoire

Synthèse combinatoire – 4- Circuit intégrée (full custom, ASIC)



ASIC (Application Specific Integrated Circuit)

circuit dédié
- ⇨ plus performant
- ⇨ coût réduit pour des grandes quantités
- ⇨ temps de conception plus long
- ⇨ pas flexible



52

52

ENSICAEN
Digital Electronic
III. Logique combinatoire

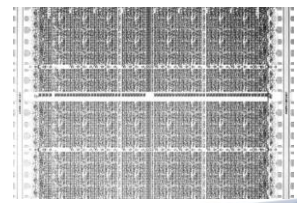
Choix de la cible pour l'implémentation

spécifique versus flexible

spécifique versus flexible

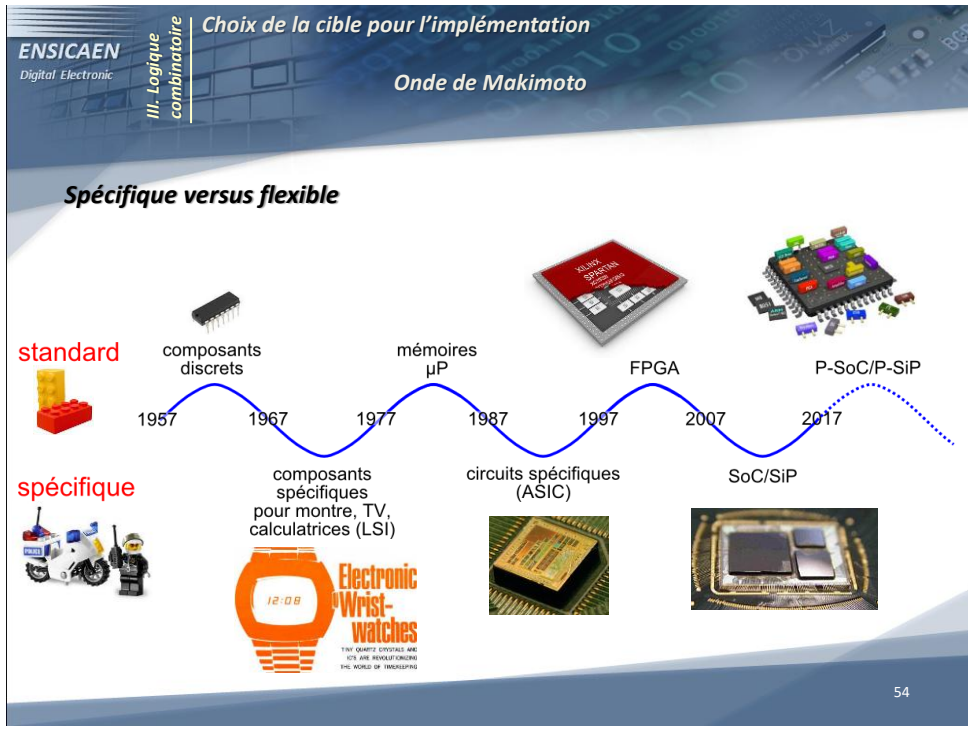
- Permanent, ne peut plus être modifié
- Temps de conception long (plrs mois à > d'un an)
- *Non-recurring engineering (NRE) costs (10^5-10^6 $)*
- Optimisé pour l'application (meilleures performances -vitesse, consommation-)
- Coût plus faible pour de grande production

- peut être modifié (adaptation à un cahier des charges changeant)
- Temps de conception plus court (programmation rapide, test direct, moins de corps de métiers)
- Logiciels de développement peu onéreux

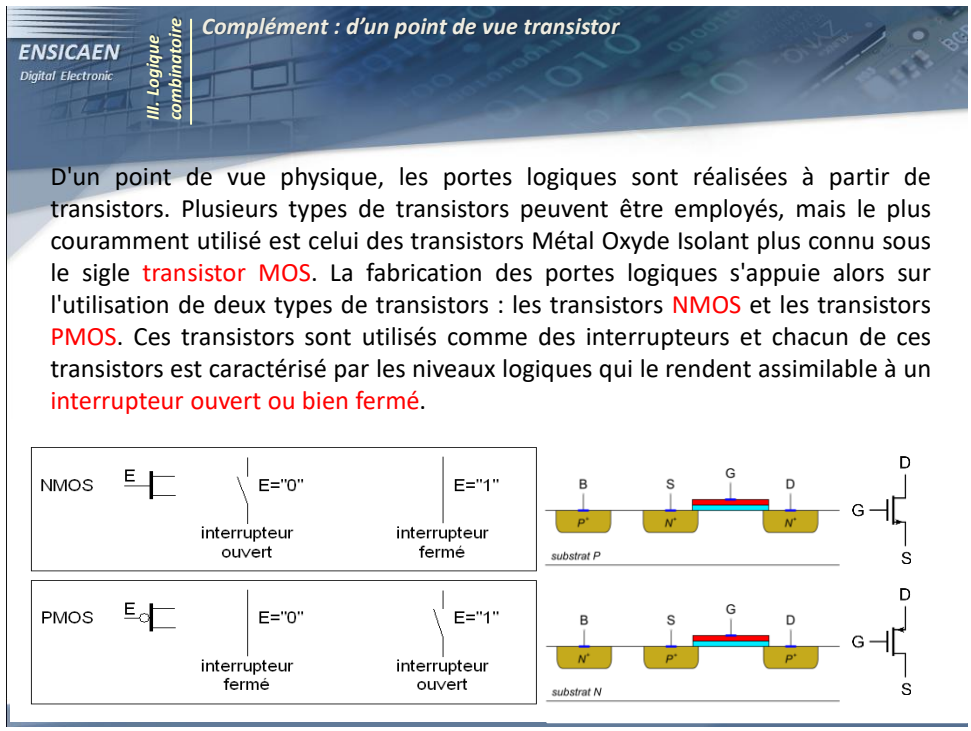


53

53


ENSICAEN
Digital Electronic
III. Logique combinatoire
Choix de la cible pour l'implémentation
Onde de Makimoto
Spécifique versus flexible
standard
spécifique
composants discrets
mémoires µP
FPGA
P-SoC/P-SiP
1957
1967
1977
1987
1997
2007
2017
composants spécifiques pour montre, TV, calculatrices (LSI)
circuits spécifiques (ASIC)
SoC/SiP
Electronic Wrist-watches
54

54

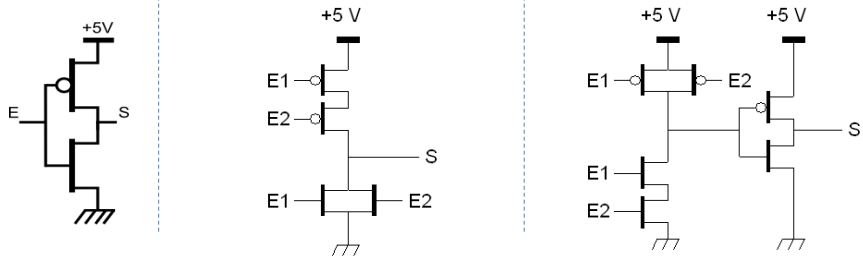

ENSICAEN
Digital Electronic
III. Logique combinatoire
Complément : d'un point de vue transistor
D'un point de vue physique, les portes logiques sont réalisées à partir de transistors. Plusieurs types de transistors peuvent être employés, mais le plus couramment utilisé est celui des transistors Métal Oxyde Isolant plus connu sous le sigle transistor MOS. La fabrication des portes logiques s'appuie alors sur l'utilisation de deux types de transistors : les transistors NMOS et les transistors PMOS. Ces transistors sont utilisés comme des interrupteurs et chacun de ces transistors est caractérisé par les niveaux logiques qui le rendent assimilable à un interrupteur ouvert ou bien fermé.
NMOS
E
E="0"
interrupteur ouvert
E="1"
interrupteur fermé
PMOS
E="0"
interrupteur fermé
E="1"
interrupteur ouvert
B
S
G
D
substrat P
substrat N

55

ENSICAEN
Digital Electronic
III. Logique combinatoire

Complément : d'un point de vue transistor (cont.)

Exemples de portes logiques



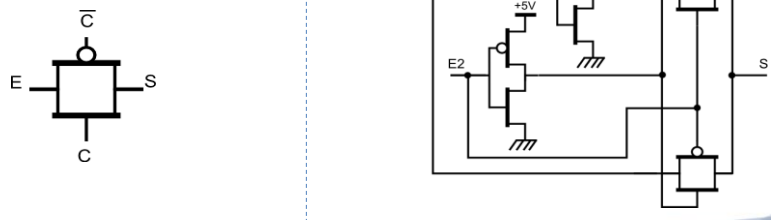
56

56

ENSICAEN
Digital Electronic
III. Logique combinatoire

Complément : d'un point de vue transistor (cont.)

Exemples de portes logiques



57

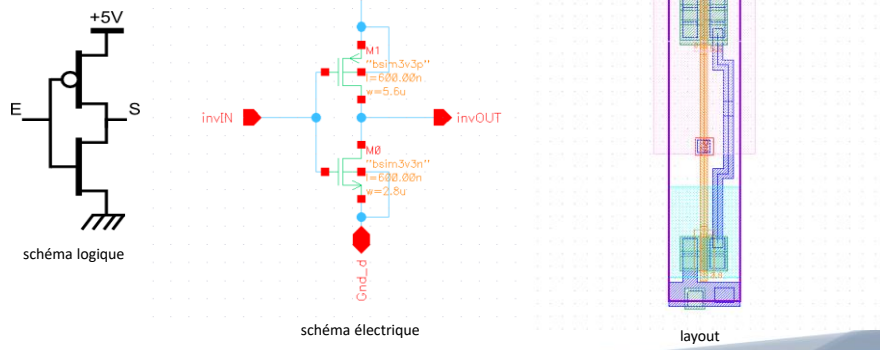
57

ENSICAEN
Digital Electronic
III. Logique combinatoire

Complément : d'un point de vue transistor (cont.)

derrière le transistor le silicium…

Exemples de layout de portes logiques



58

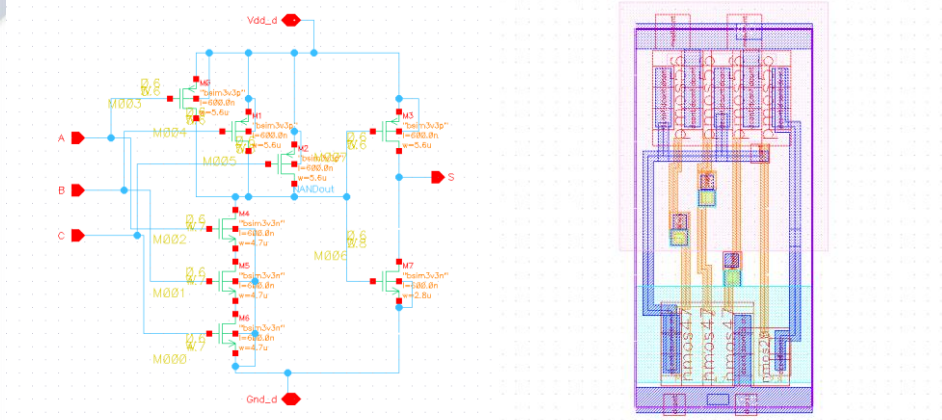
58

ENSICAEN
Digital Electronic
III. Logique combinatoire

Complément : d'un point de vue transistor (cont.)

derrière le transistor le silicium…

Exemples de layout de portes logiques (cont.)



59

59

ENSICAEN
Digital Electronic
III. Logique combinatoire

Performances des circuits numériques

Temps de commutation et puissance dissipée

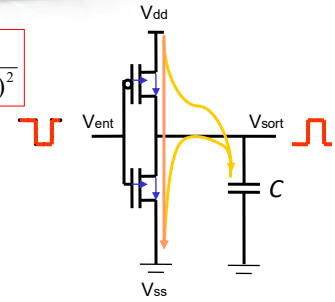
- Temps de commutation (delay)

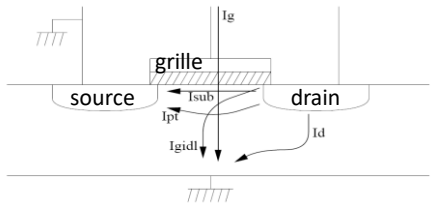
$$delay \propto \frac{V_{dd}}{(V_{dd} - V_{th})^2}$$

- Puissance dissipée
  - Deux aspects différents :
    - Puissance dynamique : activité

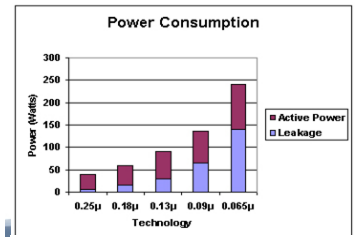
$$P = C.V_{dd}^2.F$$

    - Puissance de fuite (*leakage power*)





Isub : courant de fuite sous le seuil (*subthreshold off state leakage current*)
Ig : courant de fuite par effet tunnel (*gate tunnelling leakage current*)
Id : courant de fuite de jonction inverse (*reverse junction leakage current*)
Igidl : courant de fuite induit par la grille (*Gate Induced Drain Leakage*)
Ipt : courant de fuite transistor à canal court (*Drain Source Punch Through*)



Technologie sub-micron → 50/50

60

ENSICAEN
Digital Electronic
III. Logique combinatoire

A l'issue de cette troisième partie

- Quel outil graphique permet de faciliter la synthèse de petits circuits logiques combinatoires ? Quelle est sa limitation ?
- Comment fait-on la synthèse de circuit logique combinatoire sur les composants de type logique structurée ?
- Quels sont les avantages/inconvénients comparés des solutions programmables de type FPGA et fixe ASIC ?

61

61

ENSICAEN
Digital Electronic
III. Logique combinatoire

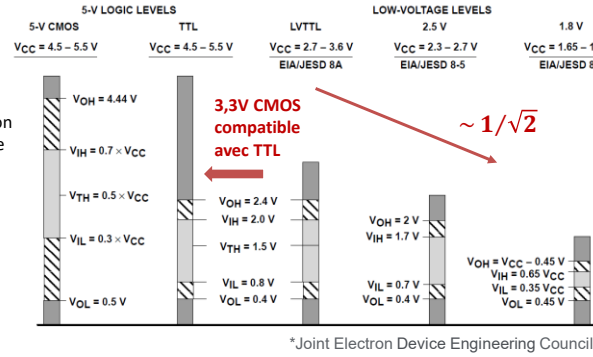
Complément : choix des tensions d'alimentation

pourquoi 5V, 3.3V, 1.8V, 0.7V ?

JEDEC* committee
JC-16 Interface Technology

The activities within JC-16's scope include the specification of power supply voltage levels for digital integrated circuits and the definition of electrical interfaces between the components of a system. The committee scope further encompasses interface protocols, modeling, simulation, testing environments, and verification.

For each V_{CC} a standard is defined, with commonly agreed-upon levels of input and output levels. Figure 1 shows the appropriate switching levels for 5-V, 3.3-V, and 2.5-V that have passed the JEDEC committee. Additionally, the 1.8-V level specification, as given in the data sheets of the SN74LVCxxxA and SN74AVCxxx devices, is shown.



*Joint Electron Device Engineering Council

EIAJ ED-5001A

2.2 Recommended operating conditions

Table 2 Recommended operating conditions

Parameter	Symbol	Normal range (3.3V nominal)	Wide range (3V nominal)	Unit
Power supply voltage	V_{DD}	3.0 ~3.6	2.7~3.6	V
Operating temperature range	Ta	Note 3	Note 3	℃

Note 3: Specified by manufacturer for various purposes, respectively.

62

62

ENSICAEN
Digital Electronic
III. Logique combinatoire

Complément : choix des tensions d'alimentation

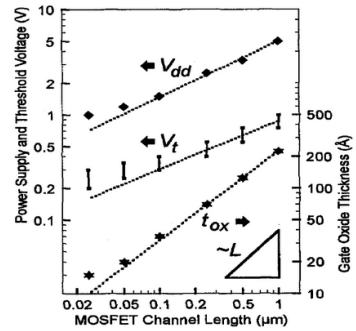
pourquoi réduit-on la tension d'alimentation ?

- Cela réduit la puissance active consommée

$$P = C.V_{dd}^2.F$$

- Pour conserver un champ électrique constant lors de la miniaturisation

$$E_{ox} = \frac{V_{dd}}{t_{ox}}$$



Évolution de la tension d'alimentation V_{dd}, de la tension de seuil V_{th} et de l'épaisseur d'oxyde en fonction de la technologie.

63

63

ENSICAEN
Digital Electronic

(Optionnel)

IV. Opérateurs arithmétiques

Principes et structures

M. Denoual

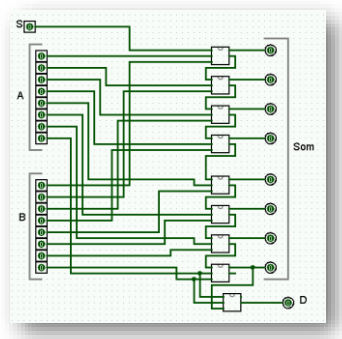
1

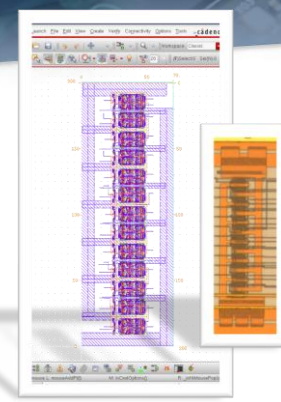
1

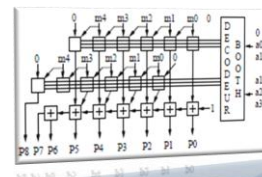
Additionneur

Soustracteur

Multiplieur







2

2

ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques

Opérateurs arithmétiques - synthèse

Cas d'un additionneur :

N nombre de bits des données

table de vérité à 2×N entrées et N+1 sorties

N+1 fonctions logiques, table à 2^{2N} lignes

AN : N=8 ⇨ 65536 de lignes

N=12 ⇨ 16,78 millions de lignes

N=16 ⇨ 4,3 milliards de lignes

On ne peut pas avoir une approche à partir des tables de vérité. Par ailleurs, une approche à partir des tables de vérité ne permettrait pas de définir les temps de propagation (temps de calcul) des opérateurs

on crée et on assemble des briques de base

Structures ordonnées, cascadables à partir de sous-structures élémentaires souvent calquées sur l'opération arithmétique élémentaire

Rmq

Remarque : on peut alors prévoir le temps de propagation (temps de calcul), la consommation de puissance et la surface (le coût)

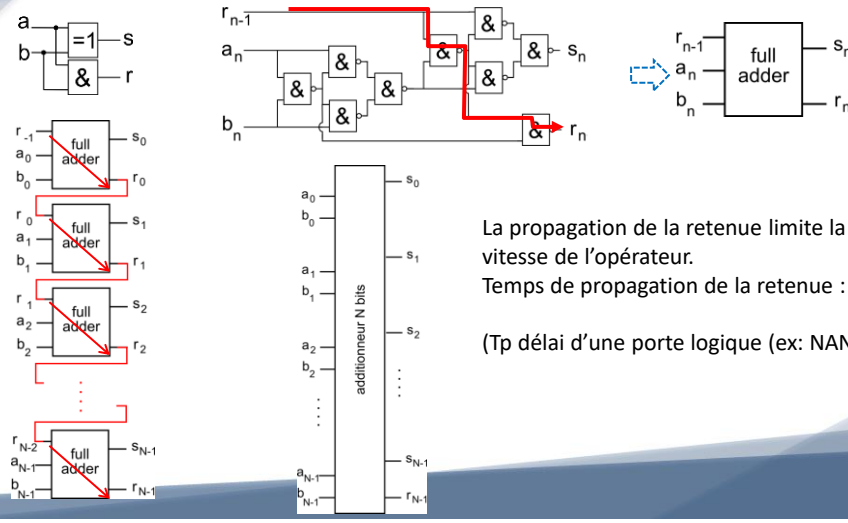
3

3

ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques

Addition – Additionneur à propagation simple

Additionneur à propagation simple (cf TD-TP)



La propagation de la retenue limite la vitesse de l'opérateur.
Temps de propagation de la retenue :

(Tp délai d'une porte logique (ex: NAND))

4

4

ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques

Addition – Additionneur à retenue anticipée

b_i	a_i	r_{i-1}	r_i	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Additionneur à retenue anticipée
(*carry look-ahead adder*)

$r_i = a_i.b_i + r_{i-1}.(a_i+b_i)$ expression de la retenue de rang i

On a une retenue au rang i : soit si l'on propage une retenue de rang i-1
soit si le full adder i génère une retenue.

On introduit :

$P_i=a_i+b_i$ propagation de rang i

$G_i=a_i.b_i$ génération de rang i

Alors :

$r_i=G_i+r_{i-1}.P_i$ expression de la retenue de rang i

Les P_i et G_i peuvent également servir à calculer la somme s_i.

$a_i \oplus b_i = G_i \oplus P_i = /G_i \,.P_i$; alors $s_i=a_i \oplus b_i \oplus r_{i-1} = G_i \oplus P_i \oplus r_{i-1} = /G_i \,.\, P_i \oplus r_{i-1}$

5

5

ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques

Addition – Additionneur à retenue anticipée (cont.)

Grâce à un circuit anticipateur de retenue, on peut calculer la retenue de sortie en 2.Tp.
Le mécanisme d'anticipation peut s'appliquer par bloc de bits.

On considère 4 fulladder 1 bits, d'indice i à i+3.

On a une retenue de dans i+3 si:

$$r_{i+3} = G_{i+3}+P_{i+3}G_{i+2}+P_{i+3}.\,P_{i+2}.G_{i+1}+P_{i+3}.\,P_{i+2}.\,P_{i+1}.G_i + P_{i+3}.P_{i+2}.P_{i+1}.P_i.r_{i-1}$$

Fonctions de génération et propagation de retenue par bloc de 4:

$$G_{i+3,i}=G_{i+3}+P_{i+3}G_{i+2}+P_{i+3}.\,P_{i+2}.G_{i+1}+P_{i+3}.\,P_{i+2}.\,P_{i+1}.G_i$$

$$P_{i+3,i}= P_{i+3}.P_{i+2}.P_{i+1}.P_i$$

$$r_{i+3}=G_{i+3,i}+ P_{i+3,i}.r_{i-1}$$

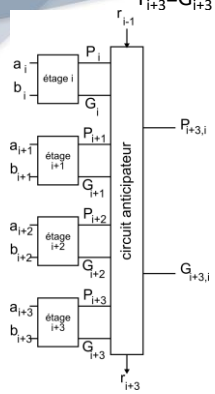
6

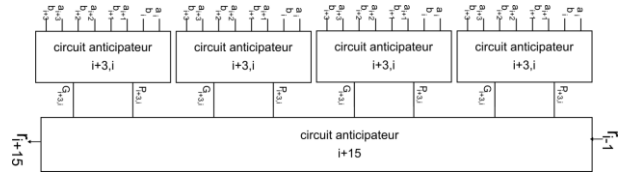
6

ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques

Addition – Additionneur à retenue anticipée (cont.)

$$r_{i+3}=G_{i+3}+P_{i+3}G_{i+2}+P_{i+3}.\ P_{i+2}.G_{i+1}+P_{i+2}.\ P_{i+2}.\ P_{i+1}.G_i+\ P_{i+3}.P_{i+2}.P_{i+1}.P_i.r_{i-1}$$





Pour un additionneur 16 bits, propagation de la retenue en 6.Tp au lieu de 16.2.Tp

$$r_{i+15}=G_{i+15,i+12}+P_{i+15,i+12}.G_{i+11,i+8}+P_{i+15,i+12}.\ P_{i+11,i+8}.G_{i+7,i+4}+P_{i+15,i+12}.\ P_{i+11,i+8}.\ P_{i+7,i+4}.G_{i+4,i+1}$$
$$+P_{i+15,i+12}.\ P_{i+11,i+8}.\ P_{i+7,i+4}.P_{i+3,i}.r_{i-1}$$

Idée : temps de propagation ne varie plus avec N (nb bits) mais avec log(N)

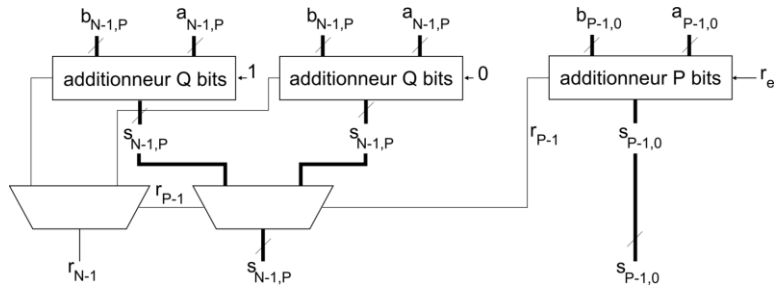
7

7

ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques

Addition – Additionneur à sélection de retenue

Additionneur à sélection de retenue



N=P+Q

Idée : les calculs sont faits en parallèle pour les deux cas possibles de retenue. Le bon résultat est sélectionné quand la retenue est connue.

8

8

ENSICAEN Digital Electronic | IV. Opérateurs arithmétiques

Addition – Comparaison des structures

- 3 types principaux d'additionneurs

	vitesse	surface	consommation
Additionneur à propagation simple	☹	🙂	-
Additionneur à retenue anticipée	🙂	-	-
Additionneur à sélection de retenue	🙂	☹	☹

9

9

ENSICAEN Digital Electronic | IV. Opérateurs arithmétiques

Soustraction

Principe

A-B = A+(-B)

Utilisation d'additionneur et d'inverseur.

Codage complément à deux, -B s'obtient par :

De manière générale : complémentation des entrées et r_0 à 1.

Additionneur/soustracteur : utilisation d'inverseurs commandés (XOR).

10

10

ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques
Additionneur-soustracteur

TEXAS INSTRUMENTS

CD54HC283, CD74HC283, CD54HCT283, CD74HCT283
SCHS176E – NOVEMBER 1997 – REVISED JULY 2022

High-Speed CMOS Logic 4-Bit Binary Full Adder with Fast Carry

1 Features

- Adds two binary numbers
- Full internal lookahead
- Fast ripple carry for economical expansion
- Operates with both positive and negative logic
- Fanout (over temperature range)
  - Standard outputs 10 LSTTL loads
  - Bus driver outputs 15 LSTTL loads
- Wide operating temperature range: –55°C to 125°C
- Balanced propagation delay and transition times
- Significant power reduction compared to LSTTL Logic ICs
- HC types

2 Description

The CDx4HC283 and CDx4HCT283 contain 4-bit binary adders. The HCT device has TTL-voltage compatible inputs.

Device Information

PART NUMBER	PACKAGE(1)	BODY SIZE (NOM)
CD54HC283	J (CDIP, 16)	24.38 mm × 6.92 mm
CD74HC283CD74HC283	D (SOIC, 16)	9.90 mm × 3.90 mm
	N (PDIP, 16)	19.31 mm × 6.35 mm
CD74HCT283CD74HCT283	D (SOIC, 16)	9.90 mm × 3.90 mm
	N (PDIP, 16)	19.31 mm × 6.35 mm

CD74HC283M96 4-bit logic adder & sutractor Hi-Speed CMOS
0,602 € Mouser 2025

11

11

ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques
UAL

Unité arithmétique et logique
alu arithmetic logic unit

pour info obsolète

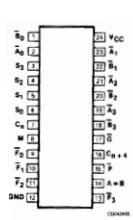
opérations arithmétiques : +,-

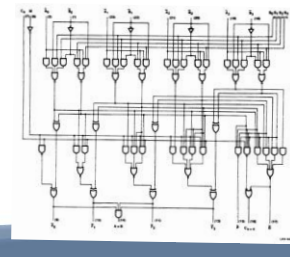
opérations logiques : ET, OU, inversion, XOR

- Exemple d'UAL : (74181TTL)

$G_i=S_3.a_i.b_i+S_2.a_i./b_i$

$P_i=a_i+S_0.b_i+S_1./b_i$





MODE SELECT — FUNCTION TABLE

MODE SELECT INPUTS				ACTIVE HIGH INPUTS & OUTPUTS	
S_3	S_2	S_1	S_0	Logic (M = H)	Arithmetic** (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	$A\overline{B}$	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*
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

MODE SELECT INPUTS				ACTIVE LOW INPUTS & OUTPUTS	
S_3	S_2	S_1	S_0	Logic (M = H)	Arithmetic** (M = L) (C_n = L)
L	L	L	L	$\overline{A}$	A minus 1
L	L	L	H	$\overline{AB}$	AB minus 1
L	L	H	L	$\overline{A+B}$	A$\overline{B}$ minus 1
L	L	H	H	Logical 1	minus 1
L	H	L	L	$\overline{A+B}$	A plus (A + $\overline{B}$)
L	H	L	H	$\overline{B}$	AB plus (A + $\overline{B}$)
L	H	H	L	$\overline{A \oplus B}$	A minus B minus 1
L	H	H	H	A + $\overline{B}$	A + $\overline{B}$
H	L	L	L	$\overline{AB}$	A plus (A + B)
H	L	L	H	$\overline{A \oplus B}$	A plus B
H	L	H	L	B	A$\overline{B}$ (A + B)
H	L	H	H	A + B	A + B
H	H	L	L	Logical 0	A plus A*
H	H	L	H	A$\overline{B}$	AB plus A
H	H	H	L	AB	AB plus A
H	H	H	H	A	A

L = LOW voltage
H = HIGH voltage level

12

12

ENSICAEN Digital Electronic | IV. Opérateurs arithmétiques

Multiplication – Multiplication combinatoire

Multiplication combinatoire

structure calquée sur la structure du calcul

		x_3	x_2	x_1	x_0		
	×	y_3	y_2	y_1	y_0		
		x_3y_0	x_2y_0	x_1y_0	x_0y_0		
	x_3y_1	x_2y_1	x_1y_1	x_0y_1			
x_3y_2	x_2y_2	x_1y_2	x_0y_2				
x_3y_3	x_2y_3	x_1y_3	x_0y_3				
p_7	p_6	p_5	p_4	p_3	p_2	p_1	p_0

$$\begin{array}{cccc} & x_3 & x_2 & x_1 & x_0 \\ \times & y_3 & y_2 & y_1 & y_0 \\ \hline \end{array}$$

13

13

ENSICAEN Digital Electronic | IV. Opérateurs arithmétiques

Multiplication – Multiplication combinatoire (cont.)

- Fait intervenir 2 types d'opérations élémentaires :
  - des produits de bits
  - des additions de sommes partielles

		x_3	x_2	x_1	x_0		
	×	y_3	y_2	y_1	y_0		
		x_3y_0	x_2y_0	x_1y_0	x_0y_0		
	x_3y_1	x_2y_1	x_1y_1	x_0y_1			
x_3y_2	x_2y_2	x_1y_2	x_0y_2	r_1			
x_3y_3	x_2y_3	x_1y_3	x_0y_3				
p_7	p_6	p_5	p_4	p_3	p_2	p_1	p_0

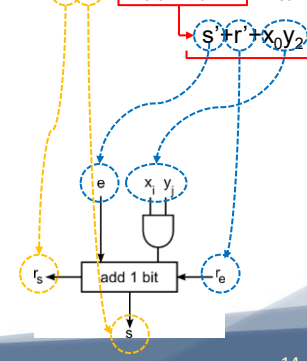
$[r_1\ p_1]=x_1y_0+x_0y_1$

$p_1=x_1y_0 \oplus x_0y_1$

$r_1=(x_1y_0).(x_0y_1)$

$[r_2\ p_2]=x_2y_0+x_1y_1+r_1+x_0y_2$

$s'+r'+x_0y_2$



- cellule élémentaire dédiée à ces opérations

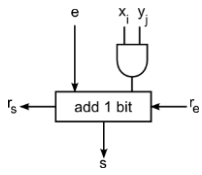
14

14

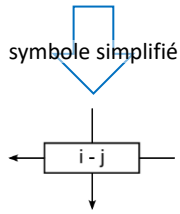
ENSICAEN Digital Electronic | IV. Opérateurs arithmétiques

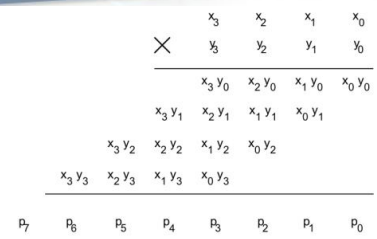
Multiplication – Multiplication combinatoire (cont.)

- Cellule élémentaire

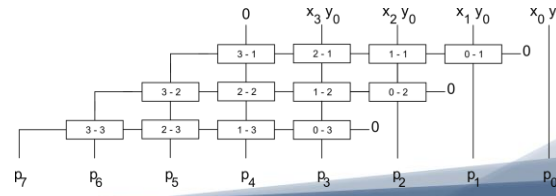


symbole simplifié





structure du multiplieur combinatoire



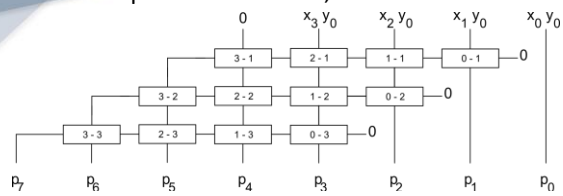
15

15

ENSICAEN Digital Electronic | IV. Opérateurs arithmétiques

Multiplication – Multiplication combinatoire (cont.)

- Exemple de calcul : X=3, Y=4



fonctionne pour les nombres non entiers en virgule fixe ? fonctionne pour les nombres signés ?

16

Multiplieur Booth

Repose sur l'algorithme de Booth, largement utilisé pour les multiplications, qui permet de réduire de moitié le nombre de sommes partielles tout en conservant une structure régulière.

Algorithme de Booth :

soit A l'opérande de multiplication : $A = -a_{N-1}\,2^{N-1} + a_{N-2}\,2^{N-2} + \dots + a_2\,2^2 + a_1\,2^1 + a_0\,2^0$

en utilisant les relations : $2^i - 2^{i-1} = 2^{i-1}$ et $2^i = 2^{i+1} - 2.2^{i-1}$

on peut écrire :

$$A = (-2.a_{N-1} + a_{N-2} + a_{N-3})2^{N-2} + (-2.a_{N-3} + a_{N-4} + a_{N-5})2^{N-4} + \dots$$
$$+ (-2.a_3 + a_2 + a_1)2^2 + (-2.a_1 + a_0 + 0)2^0$$

Il s'agit d'une décomposition de Booth d'ordre 2.

$$A = (\pm 2.a_{N-1} + a_{N-2} + a_{N-3})2^{N-2} + (-2.a_{N-3} + a_{N-4} + a_{N-5})2^{N-4} + \dots$$
$$+ (-2.a_3 + a_2 + a_1)2^2 + (-2.a_1 + a_0 + 0)2^0$$

Quand un nombre M est multiplié par A, les produits partiels obtenus sont du type :

M*-2; M*-1; M*0; M*1; M*2

Le calcul du produit final résulte de la conjugaison de 3 types de commandes :

- complémentation à 2 (*-1),
- décalage à gauche (*2),
- annulation (*0).

Multiplieur Booth constitué d'un décodeur Booth et de cellules élémentaires capables d'exécuter les 3 types de commande.

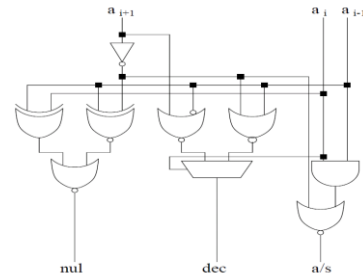
Idée : multiplication transformée en étapes logiques élémentaires (complémentation, décalage, annulation)

ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques

Multiplication – Multiplieur Booth (cont.)

- Table de vérité du décodeur Booth

a_{i+1}	a_i	a_{i-1}	produit partiel	déc. (dec)	ann. (nul)	compl. (a/s)
0	0	0	M*0	0	1	0
0	0	1	M*1	0	0	0
0	1	0	M*1	0	0	0
0	1	1	M*2	1	0	0
1	0	0	M*-2	1	0	1
1	0	1	M*-1	0	0	1
1	1	0	M*-1	0	0	1
1	1	1	M*0	0	1	0



19

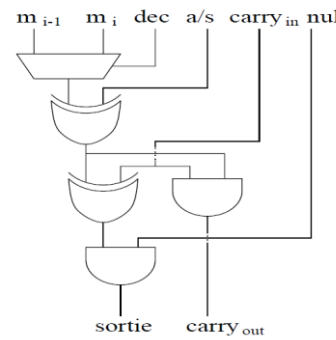
19

ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques

Multiplication – Multiplieur Booth (cont.)

- Table de vérité d'une cellule élémentaire

déc. (dec)	ann. (nul)	compl. (a/s)	sortie	retenue (carry out)
0	0	0	$m_i \oplus carry_{in}$	$m_i.carry_{in}$
0	0	1	$-m_i \oplus carry_{in}$	$-m_i.carry_{in}$
0	1	0	0	0
1	0	0	$m_{i-1} \oplus carry_{in}$	$m_{i-1}.carry_{in}$
1	0	1	$-m_{i-1} \oplus carry_{in}$	$-m_{i-1}.carry_{in}$

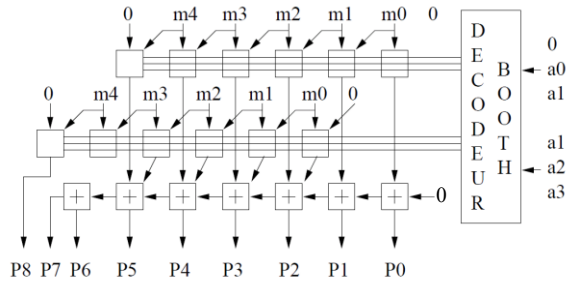


20

20

Multiplication – Multiplieur Booth (cont.)

- Structure du multiplieur Booth



réduit les propagations de retenues.

Le nombre d'opérations dépend de la configuration des bits du multiplieur.

Multiplication – Multiplieur Wallace

Multiplieur Wallace

Le multiplieur de Wallace n'a pas une structure régulière mais une structure arborescente.

Il repose sur une décomposition de la multiplication en sous-multiplications sur un nombre réduit de bits et sur la combinaison rapide des produits partiels.

Soit A et B sur N bits les nombres à multiplier. On peut écrire :

$$A*B = 2^{N}.A_{MSB}.B_{MSB}+2^{N/2}(A_{MSB}.B_{LSB}+A_{LSB}.B_{MSB})+A_{LSB}.B_{LSB}$$

Soit 4 multiplications partielles sur 2 fois moins de bits.

Idée : décomposition en sous-multiplications moins lourdes

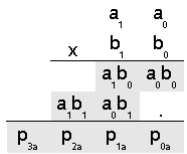
ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques

Multiplication – Multiplieur Wallace (cont.)

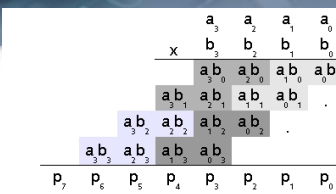
- Exemple pour N=4 bits

$A*B = 2^4.A_{MSB}.B_{MSB}+2^2(A_{MSB}.B_{LSB}+A_{LSB}.B_{MSB})+A_{LSB}.B_{LSB}$

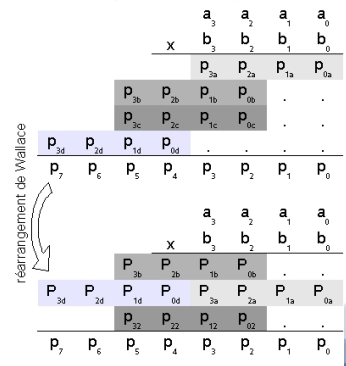
$A_{MSB}= a_3a_2$; $A_{LSB}= a_1a_0$; $B_{MSB}= b_3b_2$; $B_{LSB}= a_1a_0$



multiplication de deux nombres de 2 bits



multiplication de deux nombres de 4 bits



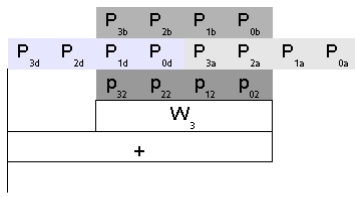
23

23

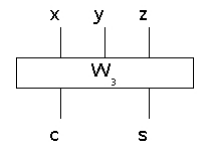
ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques

Multiplication – Multiplieur Wallace (cont.)

- Exemple pour N=4 bits



structure du multiplieur de Wallace pour la multiplication de deux nombres de 4 bits



cellule Wallace 3 : full adder

24

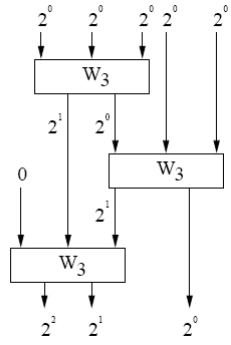
24

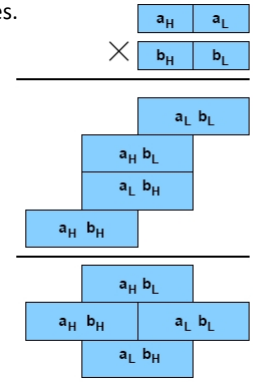
ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques

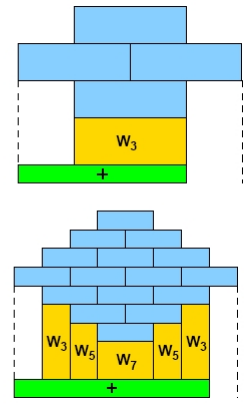
Multiplication – Multiplieur Wallace (cont.)

- N>4 bits

cellules de Wallace plus grosses.







cellule Wallace 5

25

25

ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques

Multiplication – comparaison des structures

Comparaison des multiplieurs

	temps de traversée	surface	consommation
multiplieur classique	54.70 ns	1810 (portes)	4270 µW/MHz
multiplieur Booth	42.18 ns	1020 (portes)	2195 µW/MHz

multiplieur 16 bits en technologie 0.7 µm, 3V

multiplieur Booth	5.64 ns	21 242 (µm²)	132.4 µW/MHz
multiplieur Wallace	3.28 ns	21 717 (µm²)	116.9 µW/MHz

multiplieur 16 bits en technologie 0.18 µm, 1,95V

Rmq

Remarque : *il existe d'autre type de structures de multiplieur (Dadda par exemple), optimisées pour une caractéristique particulière.*

26

26

Division - pas d'opérateur dédié

- Algorithme de division s'appuyant sur d'autres opérateurs arithmétiques
  - Requière de la logique séquentielle

Algorithme de division sans restauration :

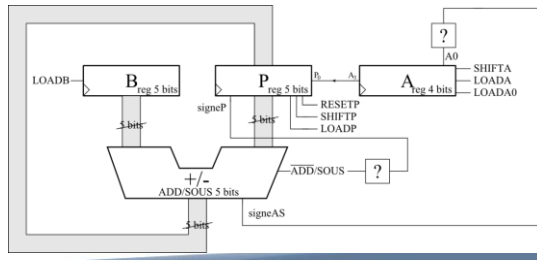
- Décalage des registres P et A de 1 bit vers la gauche.
- Si le contenu du registre P est négatif :
  - on additionne le contenu du registre B à P
- Si le contenu du registre P est positif :
  - on soustrait le contenu du registre B à P
- Si le résultat chargé dans P est négatif on met le bit de poids faible de A (A0) à « 0 » sinon à « 1 ».

Ces étapes sont répétées N fois, N étant le format des données

Exemple de l'algorithme de division sans restauration

Rmq

Remarque : comme on la fait à la main en fait

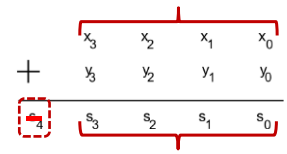


Opérateurs arithmétiques – Remarques générales - Erreurs de calculs

Rmq

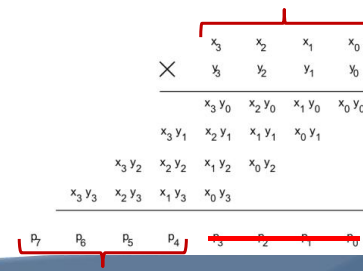
Sources d'erreur de calcul possibles

- résolution liée au format des données
  (troncature, arrondi, erreur de représentation)



- addition/soustraction :
  - débordement (*overflow*)

- multiplication :
  - arrondi/quantification



Rmq

Opérations sur des nombres non-entiers :

Cas du codage en virgule fixe :

Rappel

Rappel : la virgule n'a pas d'existence matérielle ; la représentation en virgule fixe est une interprétation du nombre binaire.
Donc pas d'impact sur les opérateurs. Ce sont les mêmes qui traitent les nombres entiers ou non-entiers à virgule fixe.

Cas du codage en virgule flottante :

Dans ce cas également les architectures d'opérateurs sont les mêmes puisque les calculs en virgule flottante se traduisent par des additions/soustractions d'entiers pour les calculs de l'exposant et par des opérations arithmétiques en virgule fixe pour la mantisse.

Note

Opérations sur des nombres signés : en complément à deux bien sûr !

add/sous : les architectures d'opérateurs vues peuvent être utilisées directement

mult : dépend de la structure

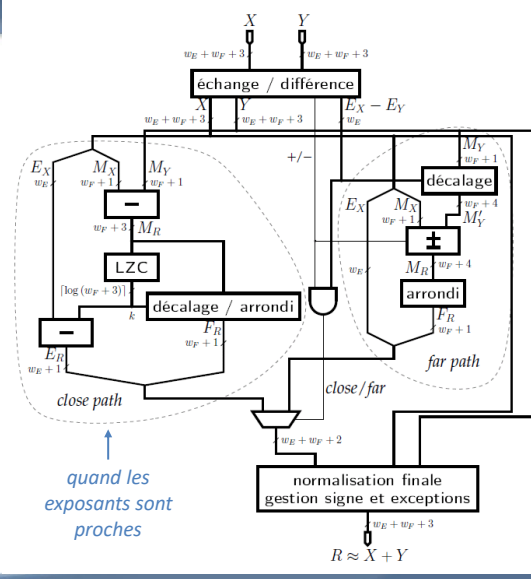
multiplication plus simple

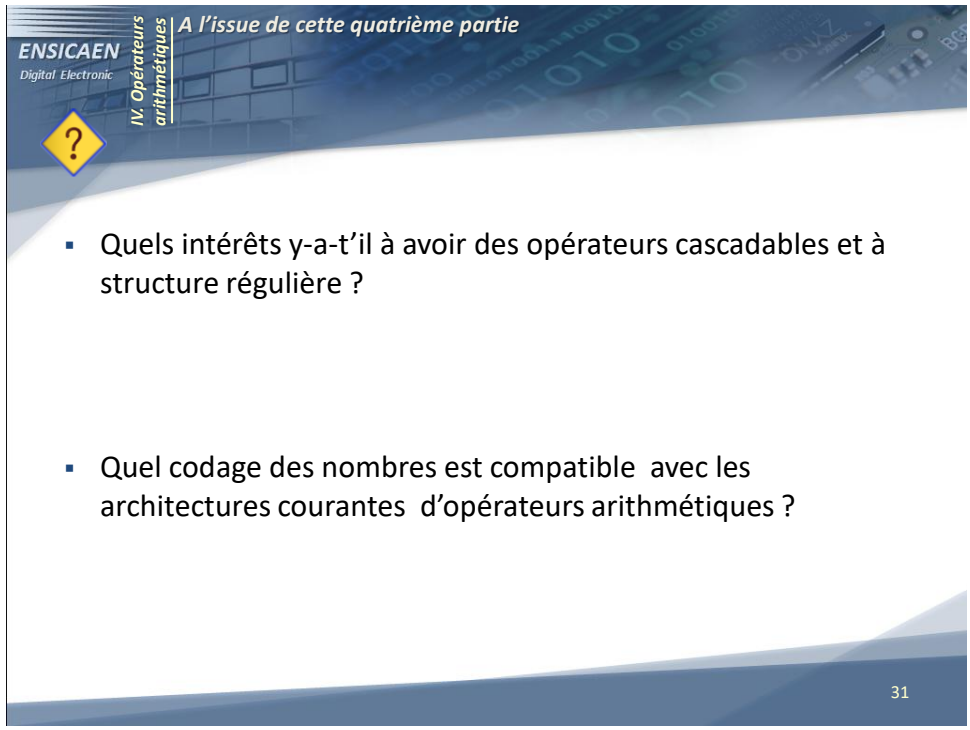
addition soustraction plus complexes.

Les nombres à additionner doivent avoir le même exposant donc il faut aligner les mantisses. C'est le nombre le plus grand qui impose la précision de calcul.

les opérateurs arithmétiques sont les mêmes que l'on soit en virgule fixe ou virgule flottante. C'est l'utilisation qu'on en fait qui est différente

Rmq

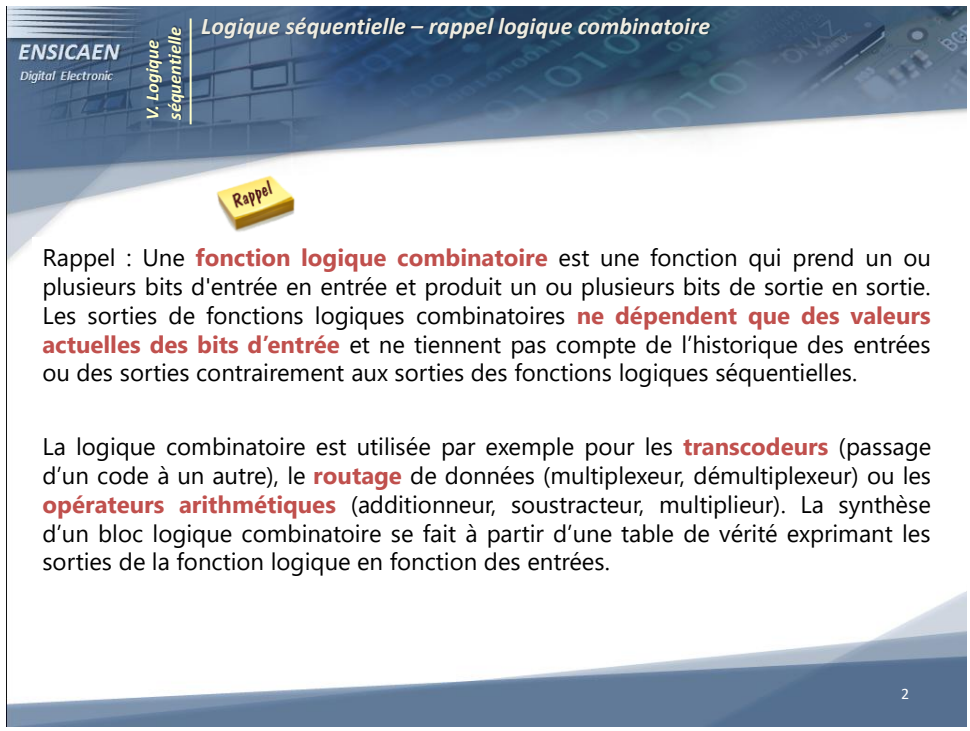



ENSICAEN
Digital Electronic
IV. Opérateurs arithmétiques
A l'issue de cette quatrième partie
Quels intérêts y-a-t'il à avoir des opérateurs cascadables et à structure régulière ?
Quel codage des nombres est compatible avec les architectures courantes d'opérateurs arithmétiques ?
31

31

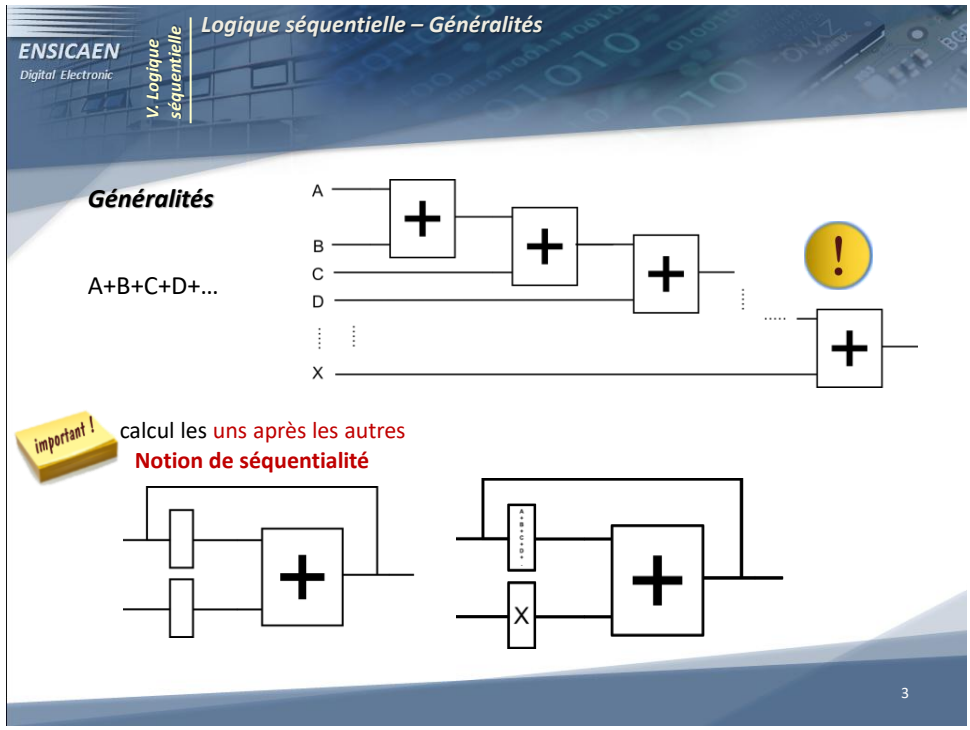

ENSICAEN
Digital Electronic
V. Logique séquentielle
Registres, compteurs, machines d'états
M. Denoual
1

1

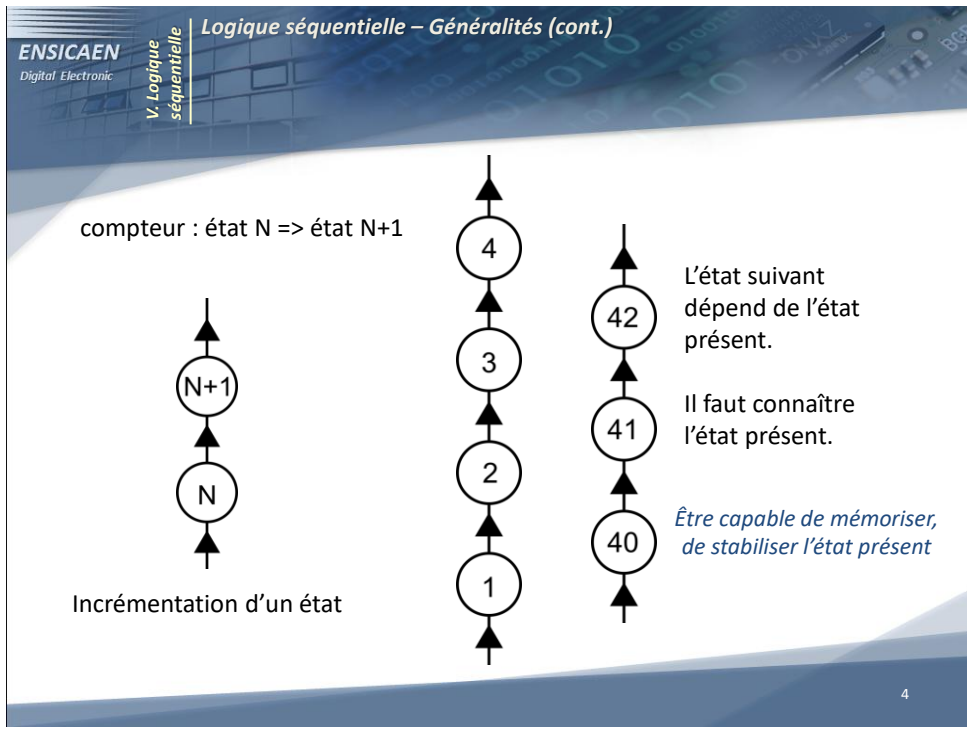

ENSICAEN
Digital Electronic
V. Logique séquentielle
Logique séquentielle – rappel logique combinatoire
Rappel
Rappel : Une fonction logique combinatoire est une fonction qui prend un ou plusieurs bits d'entrée en entrée et produit un ou plusieurs bits de sortie en sortie. Les sorties de fonctions logiques combinatoires ne dépendent que des valeurs actuelles des bits d'entrée et ne tiennent pas compte de l'historique des entrées ou des sorties contrairement aux sorties des fonctions logiques séquentielles.
La logique combinatoire est utilisée par exemple pour les transcodeurs (passage d'un code à un autre), le routage de données (multiplexeur, démultiplexeur) ou les opérateurs arithmétiques (additionneur, soustracteur, multiplieur). La synthèse d'un bloc logique combinatoire se fait à partir d'une table de vérité exprimant les sorties de la fonction logique en fonction des entrées.
2

2

ENSICAEN


ENSICAEN
Digital Electronic
V. Logique séquentielle
Logique séquentielle – Généralités
Généralités
A+B+C+D+…
A
B
C
D
X
!
important !
calcul les uns après les autres
Notion de séquentialité
X
3

3


ENSICAEN
Digital Electronic
V. Logique séquentielle
Logique séquentielle – Généralités (cont.)
compteur : état N => état N+1
N+1
N
Incrémentation d'un état
4
3
2
1
42
41
40
L'état suivant dépend de l'état présent.
Il faut connaître l'état présent.
Être capable de mémoriser, de stabiliser l'état présent
4

4

ENSICAEN
Digital Electronic
V. Logique séquentielle

Logique séquentielle – Généralités (cont.)

Les portes logiques combinatoires n'ont pas de « mémoire », la valeur de leurs sorties ne dépend que de l'état des entrées.

Ces portes adaptées au décodage ou aux opérations arithmétiques ne permettent pas seule de décrire un système dont l'état évolue.

Un état : une configuration possible du système.

Exemples :

lampe	allumée, éteinte, cassée
feu tricolore	rouge, vert, orange clignotant, hors service
ordinateur	allumé, en veille, en veille prolongé, éteint
machine à laver	prélavage, lavage, rinçage, essorage, arrêt
compteur	0, 1, 2, 3, …, N
smartphone	verrouillé, déverrouillé

des événements font évoluer le système (passage d'un état à un autre).
Exemples : signaux logiques activés (capteurs, boutons), durée écoulée (Timer)

5

5

ENSICAEN
Digital Electronic
V. Logique séquentielle

Logique séquentielle – Généralités (cont.)

Fonction logique combinatoire :

entrée interrupteur / sortie ampoule
La valeur de la sortie ne dépend que de la valeur de l'entrée.

Fonction logique séquentielle :

entrée interrupteur / sortie feu piéton
La valeur de la sortie dépend de l'état du système et de l'entrée.

Idée : en logique séquentielle, le système se souvient que l'on a appuyé et sait dans quel été il est.

6

Besoin d'un autre type de structures logiques :

portes logiques séquentielles → bascules

points mémoires

1. Bascules

Réalisées à partir de portes logiques élémentaires (ensembles de transistors).
Eléments fondamentaux permettant de mémoriser et de modifier l'information.

Utilisées pour le stockage temporaire de données et la réalisation d'automates pour le contrôle de l'évolution des systèmes.

Rmq

Remarque : *les compteurs sont un cas particulier d'automate.*

7

7

1. Bascules

Opérateurs élémentaires de mémorisation.

Leur état dépend de celui de leurs entrées mais également de leur état précédent.

état(i+1) = f(état(i), entrées)

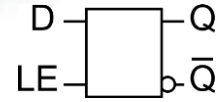
(verrouillé + touche écran => déverrouillé)

On distingue les bascules asynchrones sensibles au niveau des entrées et les bascules synchrones sensibles au front d'un signal spécial cadençant le système (typiquement un signal d'horloge).

Historiquement : bascules RS, bascules JK. A l'heure actuelle uniquement des bascules synchrones, les bascules D à front constituées de 2 bascules *latch* en série.

8

8



Bascule latch

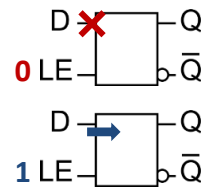
Entrée de donnée D (état) et entrée de commande (LE *latch enable*).

Q : état présent

Q+ : état suivant

Table de vérité de la bascule latch

LE	D	Q+	
0	0	Q	mode verrouillée, mémorisation
0	1	Q	
1	0	0	mode transparent
1	1	1	



Idée : soit on laisse entrer une nouvelle valeur soit on verrouille l'entrée et on mémorise le contenu.

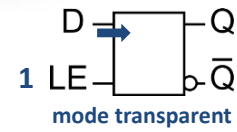
11

11

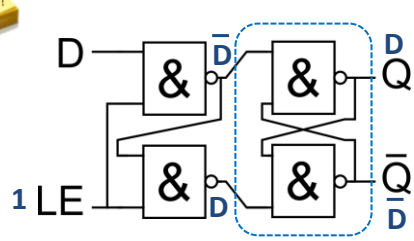
Bascule latch : structures internes

Entrée de donnée D (état) et entrée de commande (LE).

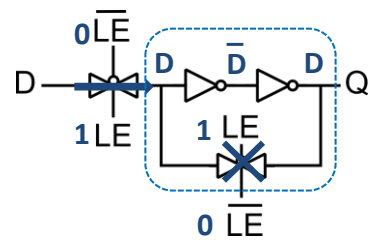
Q : état présent , Q+ : état suivant



comment ça marche ?



Structure à portes NAND



Structure compacte à portes de transmission et inverseurs

Idée : c'est le rebouclage interne qui permet de mémoriser un état.
contre-réaction qui stabilise l'état

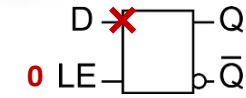
12

12

Bascule latch : structures internes

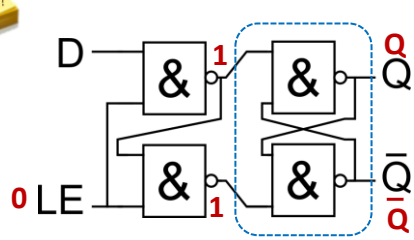
Entrée de donnée D (état) et entrée de commande (LE).

Q : état présent , Q+ : état suivant

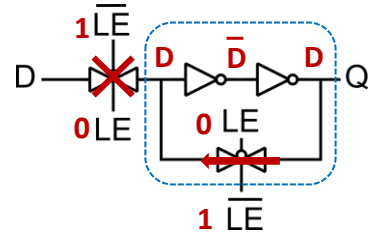


mode verrouillée, mémorisation

comment ça marche ?

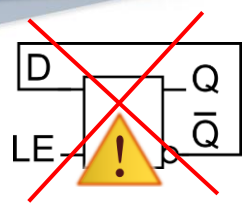


Structure à portes NAND



Structure compacte à portes de transmission et inverseurs

Idée : c'est le rebouclage interne qui permet de mémoriser un état.



pas possible avec ce type de bascule

C'est un rebouclage utilisé dans beaucoup de systèmes séquentiels (compteurs par exemple)

Compteur :

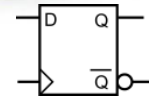
0	1	2	3	4	5
0	0	0	0	1	1
0	0	1	1	0	0
0 ➡	1 ➡	0 ➡	1 ➡	0 ➡	1

Besoin d'une structure pas entièrement transparente

ENSICAEN Digital Electronic | V. Logique séquentielle

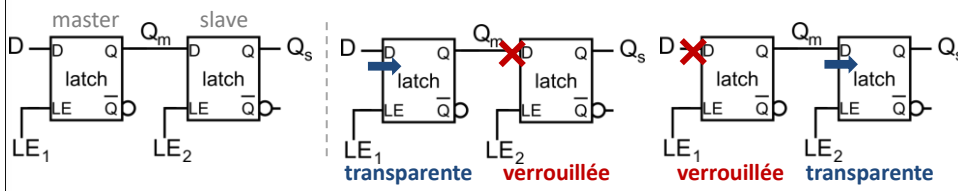
Logique séquentielle – 1. Bascules – Bascule D à front

c'est LA porte logique séquentielle dont on se sert en pratique



Bascule D à front

La bascule D est non transparente. Pas de liaison directe entre l'entrée et la sortie. Elle résulte de l'association en série de deux bascules *latch*.



Une bascule maître et une bascule esclave. Lorsque l'une est transparente, l'autre est verrouillée.
Pour garantir un fonctionnement correct, il faut des signaux de commandes LE_1 et LE_2 sans recouvrement.

15

15

ENSICAEN Digital Electronic | V. Logique séquentielle

Logique séquentielle – 1. Bascules – Bascule D (cont.)

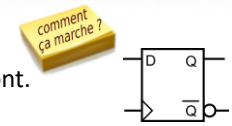
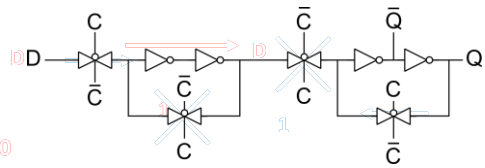
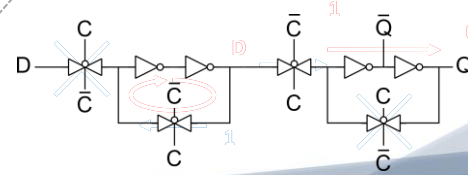


Table de vérité de la bascule D à front

D	C	Q+
0	↓	0
1	↓	1

Structure interne de la bascule D à front.





17

17

ENSICAEN
Digital Electronic
V. Logique séquentielle

Logique séquentielle – 1. Bascules – Remarques générales

***Remarques** générales en pratique :*

Rmq

Les bascules comprennent des entrées supplémentaires.
Il faut respecter des temps de mise en place et de maintien des niveaux logiques pour garantir un bon fonctionnement des bascules.

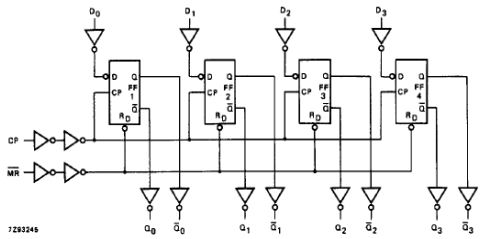


Fig.5 Logic diagram.

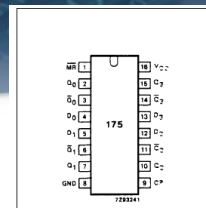


Fig.1 Pin configuration.

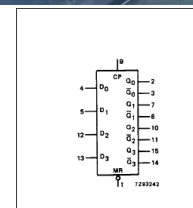
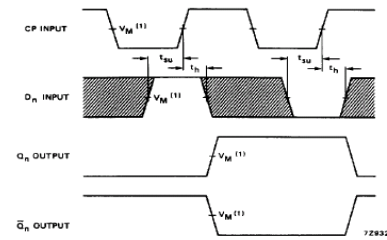


Fig.2 Logic symbol.



The shaded areas indicate when the input is permitted to change for predictable output performance.

(1) HC : V_M = 50%; V_I = GND to V_{CC}.
HCT : V_M = 1.3 V; V_I = GND to 3 V.

Fig.7 Waveforms showing the data set-up and hold times for the data input (D_n).

datasheet 74HC175 Edge Triggered Flip-Flop

18

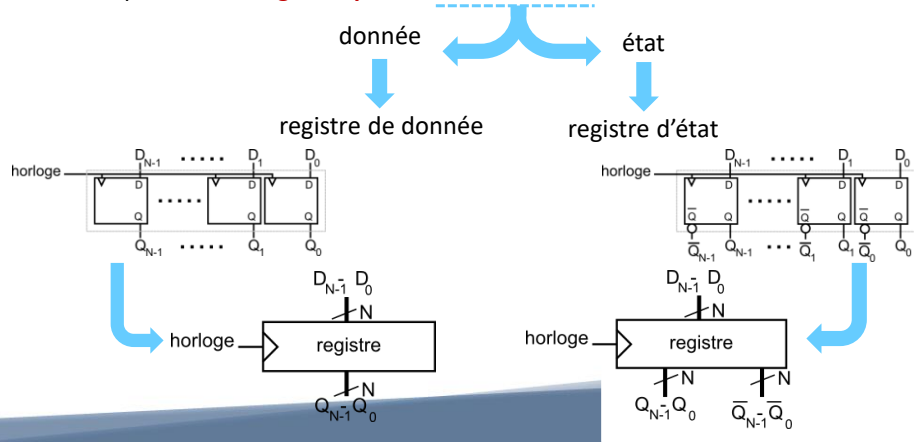
ENSICAEN
Digital Electronic
V. Logique séquentielle

Logique séquentielle – 2. Registres

Registres

registre : ensemble de bascules

pour le **stockage temporaire d'information**

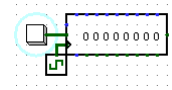


19

Logique séquentielle – 2. Registres (cont.)

Autres types de registres

- registre à chargement parallèle
- registre à décalage



- conversion série-parallèle
- conversion parallèle-série
- génération SBPA

Rmq

les données sont traitées sous forme parallèle par les opérateurs arithmétiques.
Les données lors de communications entre équipements peuvent être sous forme série

?

quelles communications série connaissez-vous ?

Logique séquentielle – 3. Machines à états finis

3. Machines à états finis (MAEF)

Finite State Machine (FSM) ou automates synchrones.

Une machine d'état est un opérateur séquentiel dont la sortie est fonction des entrées et de l'état précédent de la machine d'état. Elle utilise des bascules pour mémoriser l'état présent et des blocs de logique combinatoire pour générer les sorties et l'état futur.

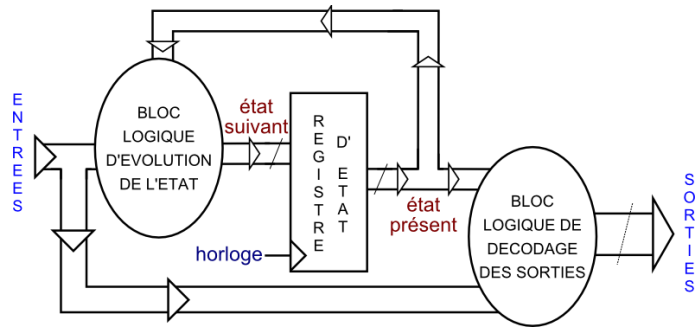
Il existe 2 types de structure de machine d'état:

1. Machine de Mealy : les sorties dépendent de l'état et des entrées
2. Machine de Moore : les sorties dépendent exclusivement de l'état

Rmq

Remarque : *les compteurs sont un cas particulier de machine à états finis.*

3.1 Machines à états finis (MAEF) type Mealy



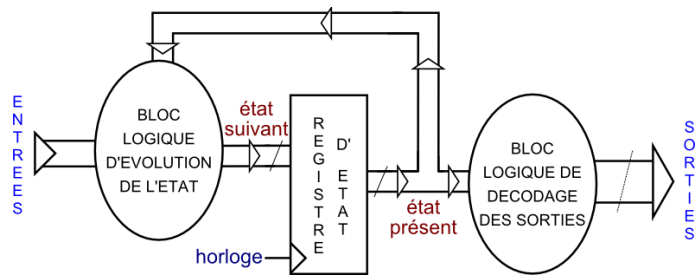
les sorties peuvent changer de manière asynchrone.

À éviter si non indispensable

22

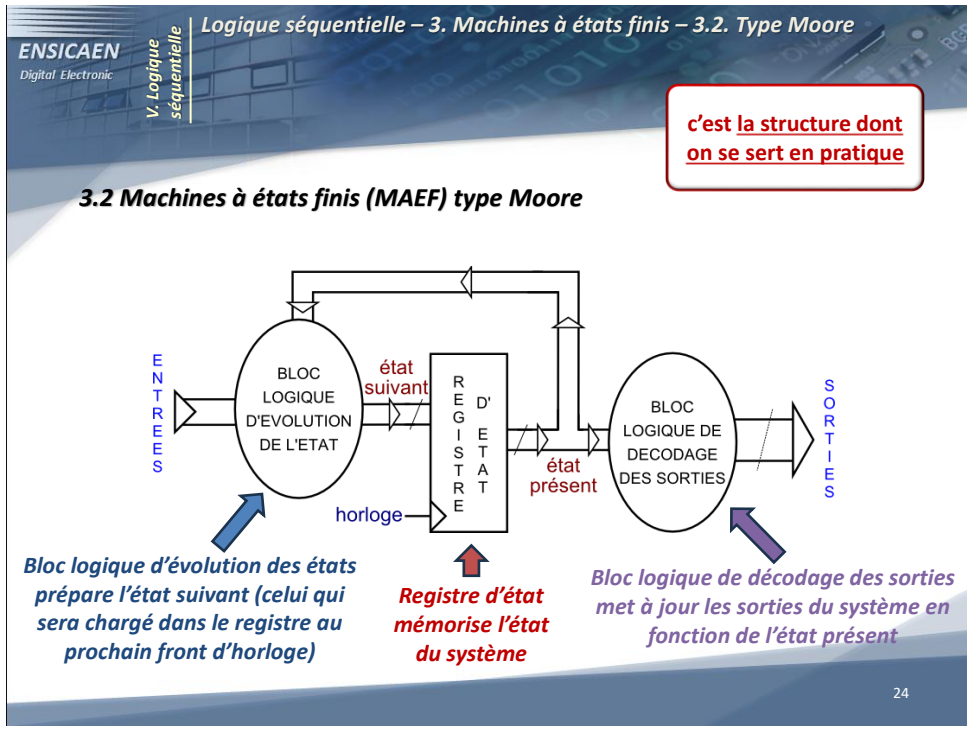
c'est la structure dont on se sert en pratique

3.2 Machines à états finis (MAEF) type Moore

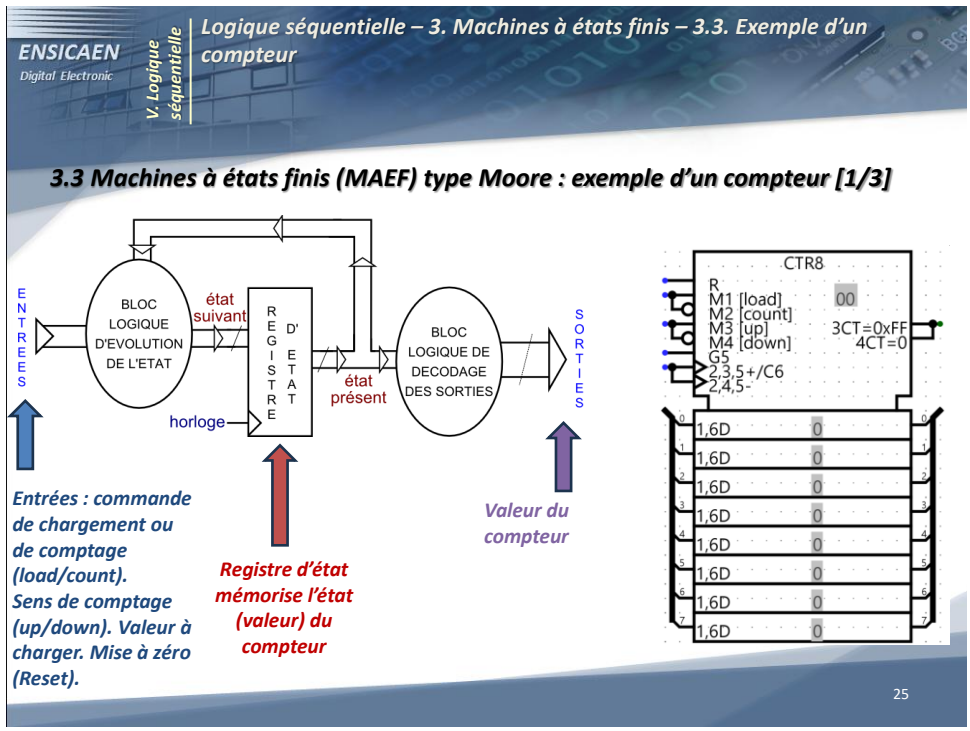


Le type Moore garantit des **signaux logiques synchrones**. Il est privilégié dans les applications intégrées et cœur de calculateur pour le séquenceur.

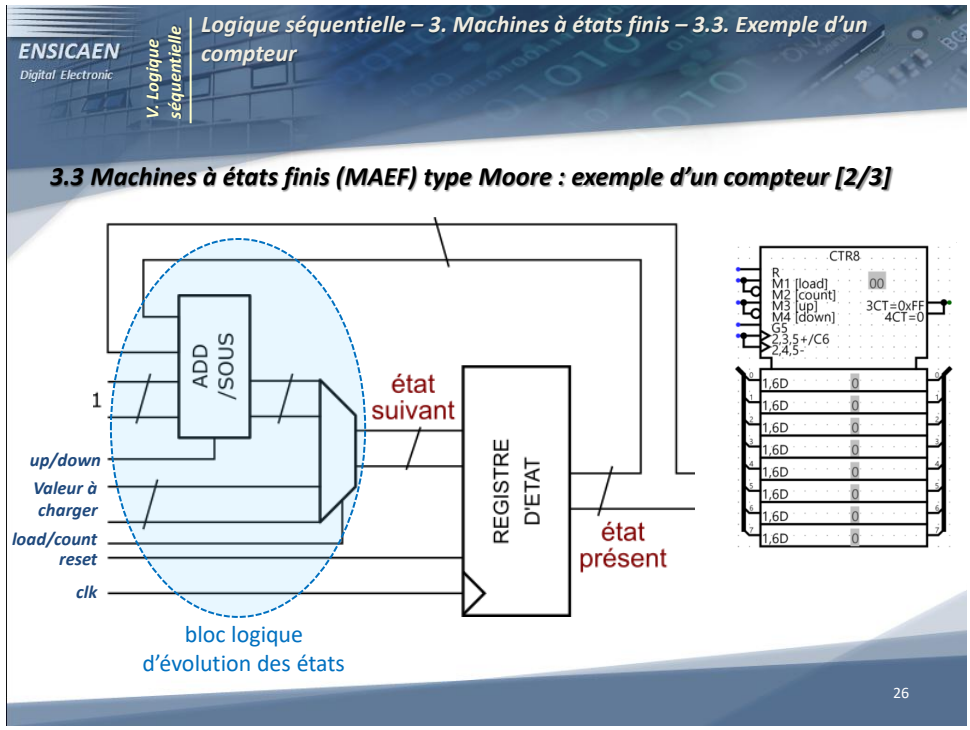
23


ENSICAEN
Digital Electronic
V. Logique séquentielle
Logique séquentielle – 3. Machines à états finis – 3.2. Type Moore
c'est la structure dont on se sert en pratique
3.2 Machines à états finis (MAEF) type Moore
ENTREES
BLOC LOGIQUE D'EVOLUTION DE L'ETAT
état suivant
REGISTRE D'ETAT
état présent
BLOC LOGIQUE DE DECODAGE DES SORTIES
SORTIES
horloge
Bloc logique d'évolution des états prépare l'état suivant (celui qui sera chargé dans le registre au prochain front d'horloge)
Registre d'état mémorise l'état du système
Bloc logique de décodage des sorties met à jour les sorties du système en fonction de l'état présent
24

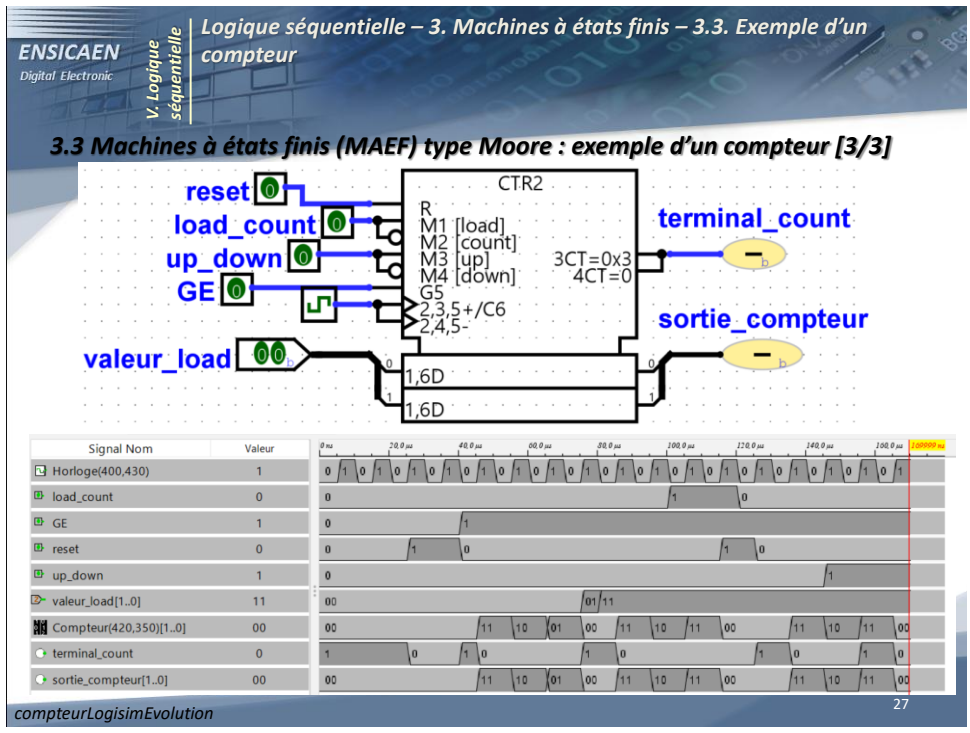
24


ENSICAEN
Digital Electronic
V. Logique séquentielle
Logique séquentielle – 3. Machines à états finis – 3.3. Exemple d'un compteur
3.3 Machines à états finis (MAEF) type Moore : exemple d'un compteur [1/3]
ENTREES
BLOC LOGIQUE D'EVOLUTION DE L'ETAT
état suivant
REGISTRE D'ETAT
état présent
BLOC LOGIQUE DE DECODAGE DES SORTIES
SORTIES
horloge
CTR8
R
M1 [load]
M2 [count]
M3 [up]
M4 [down]
G5
2,3,5+/C6
2,4,5-
00
3CT=0xFF
4CT=0
1,6D
Entrées : commande de chargement ou de comptage (load/count). Sens de comptage (up/down). Valeur à charger. Mise à zéro (Reset).
Registre d'état mémorise l'état (valeur) du compteur
Valeur du compteur
25

25


ENSICAEN
Digital Electronic
V. Logique séquentielle
Logique séquentielle – 3. Machines à états finis – 3.3. Exemple d'un compteur
3.3 Machines à états finis (MAEF) type Moore : exemple d'un compteur [2/3]
ADD /SOUS
1
up/down
Valeur à charger
load/count
reset
clk
état suivant
REGISTRE D'ETAT
état présent
CTR8
R
M1 [load]
M2 [count]
M3 [up]
M4 [down]
G5
2,3,5+/C6
2,4,5-
00
3CT=0xFF
4CT=0
1,6D
bloc logique d'évolution des états
26

26

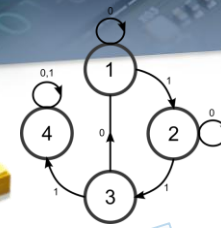

ENSICAEN
Digital Electronic
V. Logique séquentielle
Logique séquentielle – 3. Machines à états finis – 3.3. Exemple d'un compteur
3.3 Machines à états finis (MAEF) type Moore : exemple d'un compteur [3/3]
reset
load_count
up_down
GE
valeur_load
CTR2
R
M1 [load]
M2 [count]
M3 [up]
M4 [down]
G5
2,3,5+/C6
2,4,5-
3CT=0x3
4CT=0
1,6D
terminal_count
sortie_compteur
Signal Nom
Valeur
Horloge(400,430)
load_count
GE
reset
up_down
valeur_load[1..0]
Compteur(420,350)[1..0]
terminal_count
sortie_compteur[1..0]
compteurLogisimEvolution
27

27

3.4 Machines à états finis (MAEF), synthèse

Méthode

à retenir !



En **6 étapes** :

1. **déterminer le diagramme d'état en fonction du cahier des charges**

identifier les entrées, les sorties et le nombre d'états.

binaire, Gray, One hot, … ?

2. **choisir un codage pour la représentation des états** ***(par défaut binaire)***
3. **établir la table d'évolution des états**
4. **synthétiser les blocs logiques d'évolution des états**
5. **établir la table de vérité des sorties**
6. **synthétiser le bloc logique de décodage des sorties**

entrées	état présent	état suivant
xxxx	xxxx	xxxx
xxxx	xxxx	xxxx
xxxx	xxxx	xxxx

BLOC LOGIQUE D'EVOLUTION DE L'ETAT

état présent	sorties
xxxx	xx
xxxx	xx
xxxx	xx

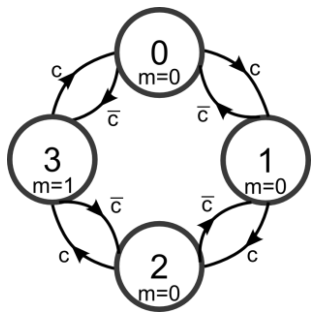
BLOC LOGIQUE DE DECODAGE DES SORTIES

28

28

Exemple : compteur-décompteur 4, avec m actif pour la valeur maximum

1. diagramme d'états
une entrée c, une sortie m, 4 états



c=1 : comptage
c=0 : décomptage

2. choix d'un codage pour les états
ici exemple avec binaire, Gray, one hot

	c	état présent		état suivant	
comptage	1	0	xxxx	1	xxxx
	1	1	xxxx	2	xxxx
	1	2	xxxx	3	xxxx
	1	3	xxxx	0	xxxx
décomptage	0	0	xxxx	3	xxxx
	0	1	xxxx	0	xxxx
	0	2	xxxx	1	xxxx
	0	3	xxxx	2	xxxx

29

29

3. établir la table d'évolution des états

Exemple : compteur-décompteur 4

table d'évolution avec **codage binaire**

c	état présent			état suivant		
		Q1	Q0		D1	D0
1	0	0	0	1	0	1
1	1	0	1	2	1	0
1	2	1	0	3	1	1
1	3	1	1	0	0	0
0	0	0	0	3	1	1
0	1	0	1	0	0	0
0	2	1	0	1	0	1
0	3	1	1	2	1	0

30

4. synthétiser le bloc logique d'évolution des états

Exemple : compteur-décompteur 4

table d'évolution avec **codage binaire**

c	état présent			état suivant		
		Q1	Q0		D1	D0
1	0	0	0	1	0	1
1	1	0	1	2	1	0
1	2	1	0	3	1	1
1	3	1	1	0	0	0
0	0	0	0	3	1	1
0	1	0	1	0	0	0
0	2	1	0	1	0	1
0	3	1	1	2	1	0

31

5. établir la table de vérité des sorties

Exemple : compteur-décompteur 4

avec **codage binaire**

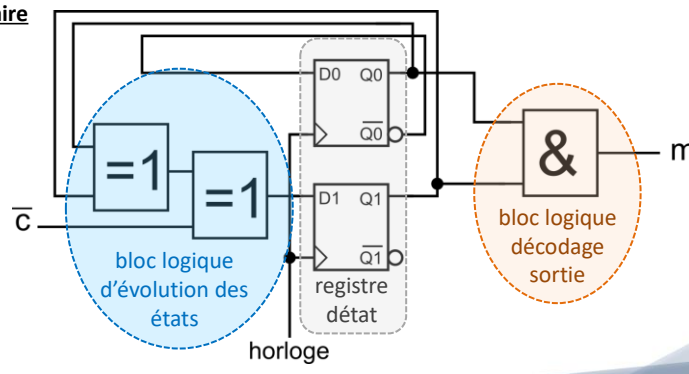
	état présent		m
	Q1	Q0	
0	0	0	0
1	0	1	0
2	1	0	0
3	1	1	1

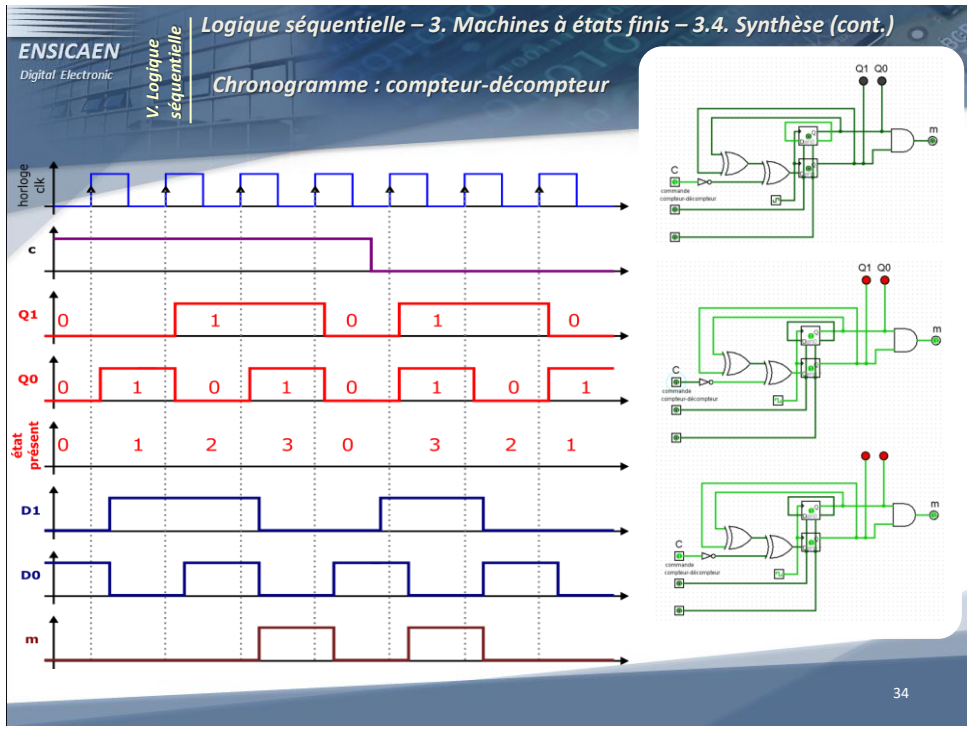
ne dépend que de l'état présent (machine de type Moore)

6. synthétiser le bloc logique de décodage des sorties

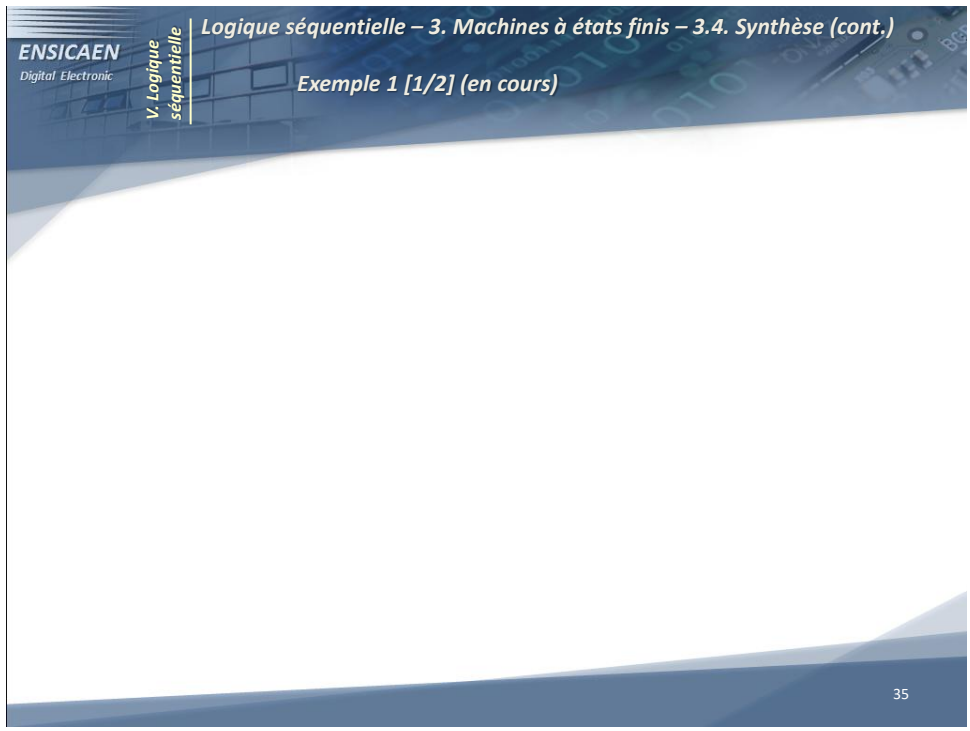
Exemple : compteur-décompteur 4

avec **codage binaire**




ENSICAEN
Digital Electronic
V. Logique séquentielle
Logique séquentielle – 3. Machines à états finis – 3.4. Synthèse (cont.)
Chronogramme : compteur-décompteur
horloge clk
c
Q1
Q0
état présent
D1
D0
m
0 1 2 3 0 3 2 1
Q1 Q0
C
commande compteur-décompteur
34

34


ENSICAEN
Digital Electronic
V. Logique séquentielle
Logique séquentielle – 3. Machines à états finis – 3.4. Synthèse (cont.)
Exemple 1 [1/2] (en cours)
35

35

ENSICAEN
Digital Electronic
V. Logique séquentielle

Logique séquentielle – 3. Machines à états finis – 3.4. Synthèse (cont.)

Exemple 1 [2/2] (en cours)

36

36

ENSICAEN
Digital Electronic
V. Logique séquentielle

Logique séquentielle – 3. Machines à états finis – 3.4. Synthèse (cont.)

Exemple synthèse 2 [1/3]

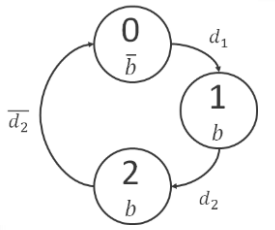
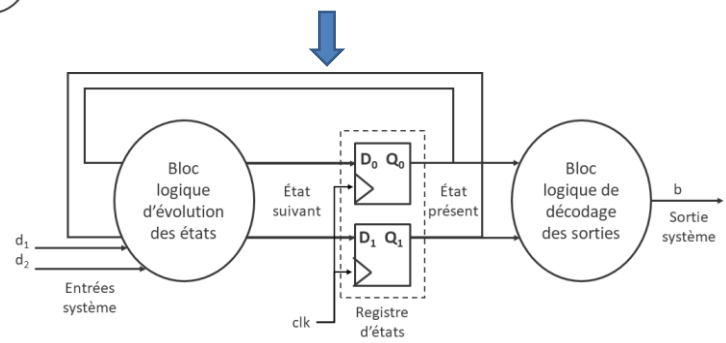


Diagramme d'état

3 états → 2 bascules

Machine d'état à 3 états, 2 entrées d_1 et d_2 et 1 sortie b.



37

37

ENSICAEN Digital Electronic | V. Logique séquentielle

Logique séquentielle – 3. Machines à états finis – 3.4. Synthèse (cont.)

Exemple synthèse 2 [2/3]

Table de vérité du bloc logique d'évolution des états

entrées					sorties		
Entrées systèmes		État présent			État suivant		
d_1	d_2	Q_1	Q_0	État prés.	État suiv.	D_1	D_0
0	X	0	0	0	0	0	0
1	X	0	0	0	1	0	1
X	0	0	1	1	1	0	1
X	1	0	1	1	2	1	0
X	1	1	0	2	2	1	0
X	0	1	0	2	0	0	0
X	X	1	1	3	0*	0	0

*Intérêt à renvoyer vers un cycle du système, sinon reste planté.
Mais cela dépend des applications et cahiers des charges.

Table de Karnaugh pour D_1

$d_1 d_2$ \ Q_1Q_0	00	01	11	10
00	0	0	0	0
01	0	1	0	1
11	0	1	0	1
10	0	0	0	0

Table de Karnaugh pour D_0

$d_1 d_2$ \ Q_1Q_0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	0	0
10	1	1	0	0

38

38

ENSICAEN Digital Electronic | V. Logique séquentielle

Logique séquentielle – 3. Machines à états finis – 3.4. Synthèse (cont.)

Exemple synthèse 2 [3/3]

Les tables de Karnaugh ne permettent pas de grande simplification dans ce cas.

$$D_1 = d_2 \cdot (Q_1 \oplus Q_0)$$

$$D_0 = d_1 \cdot \overline{Q_1} \cdot \overline{Q_0} + \overline{d_2} \cdot \overline{Q_1} \cdot Q_0$$

Table de vérité du bloc logique de décodage des sorties

État présent (entrée fonction)			sortie
Q_1	Q_0	état	b
0	0	0	0
0	1	1	1
1	0	2	1
1	1	3	X

$$(b = Q_1 + Q_0)$$
$$b = Q_1 \oplus Q_{0_2}$$

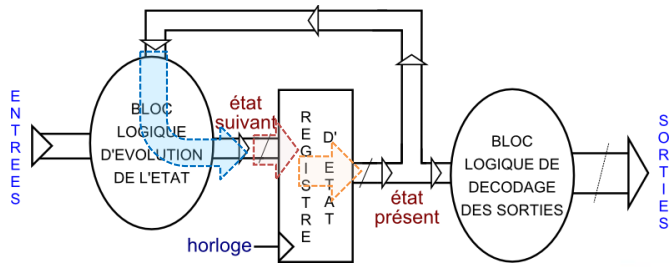
39

39

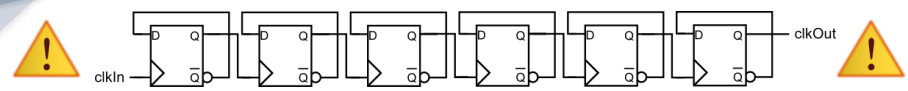
Remarque : *fréquence de fonctionnement, cadencement de l'horloge*

Rmq

$$F_{\max} = \frac{1}{T_{\text{horloge}_{\min}}} = \frac{1}{t_{p_{\text{bascule}}} + t_{p_{\text{bloc logique}}} + t_{\text{set-up}}}$$

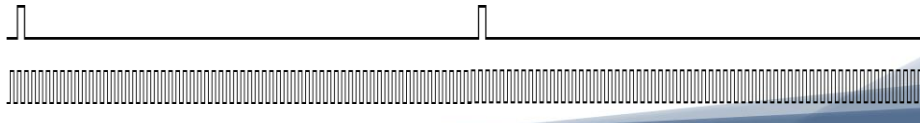


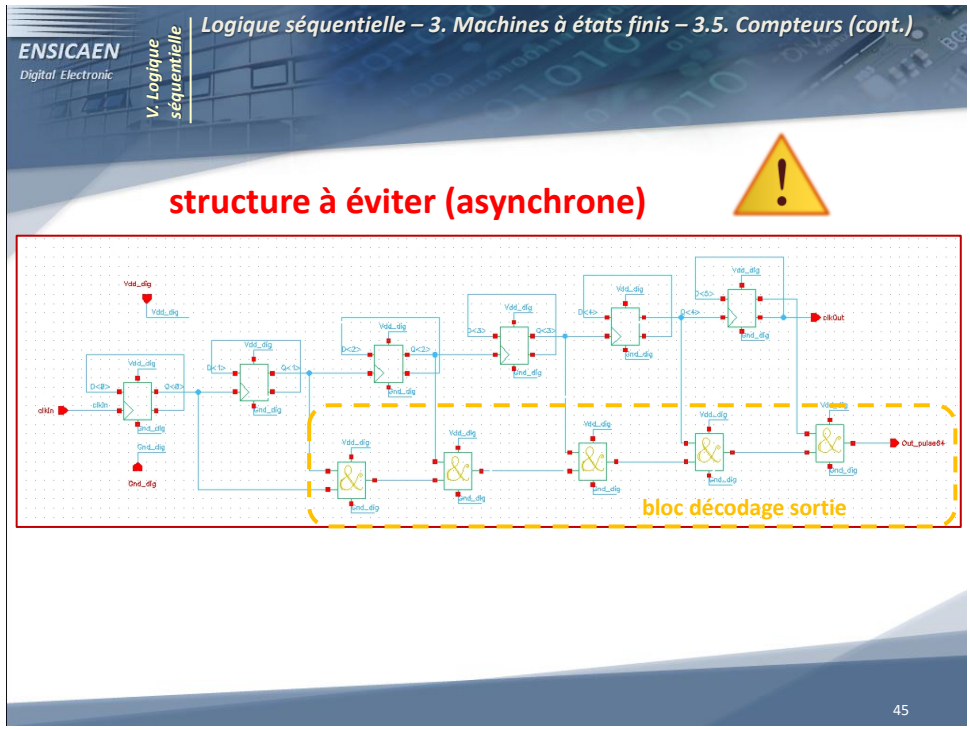
Structure à éviter



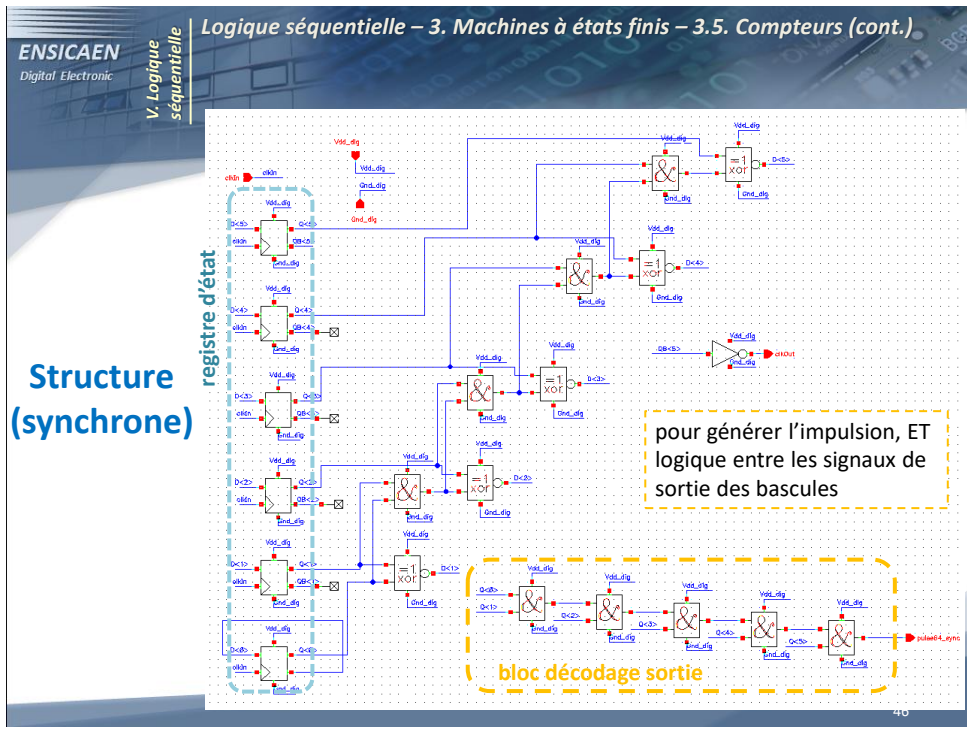
Les signaux de sortie des bascules de cette structure ne sont pas synchrones. On ne peut pas s'en servir pour générer des signaux à cause des aléas (états transitoires non souhaités).

Exemple : on souhaite générer une impulsion toutes les 64 périodes (fin de comptage (cf Timer 0 du PIC18)).

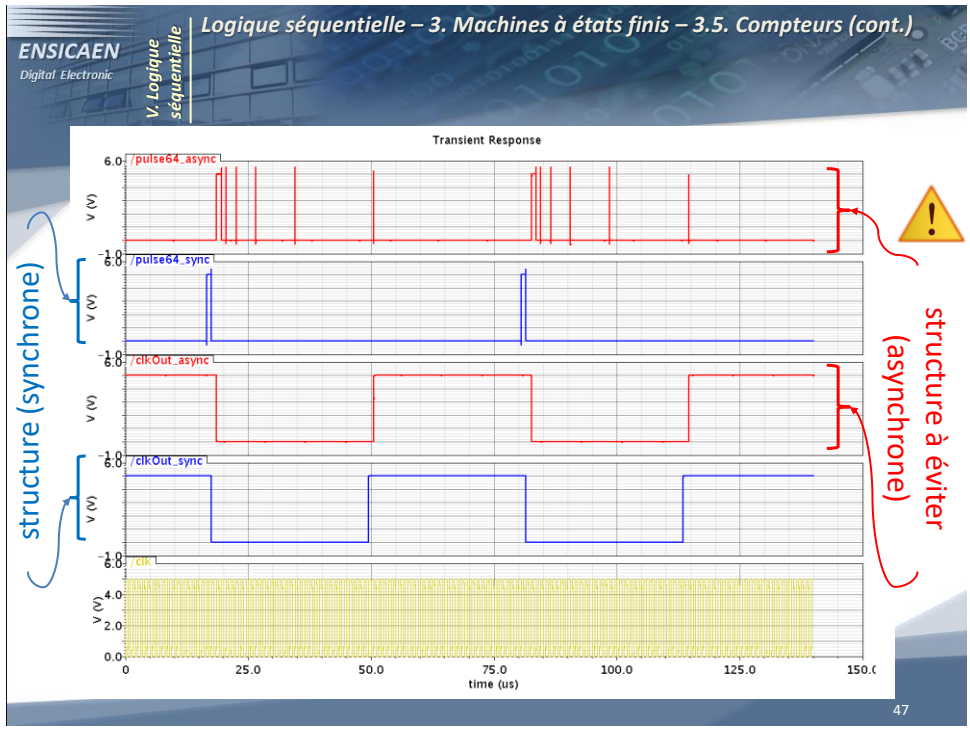



ENSICAEN
Digital Electronic
V. Logique séquentielle
Logique séquentielle – 3. Machines à états finis – 3.5. Compteurs (cont.)
structure à éviter (asynchrone)
bloc décodage sortie
45

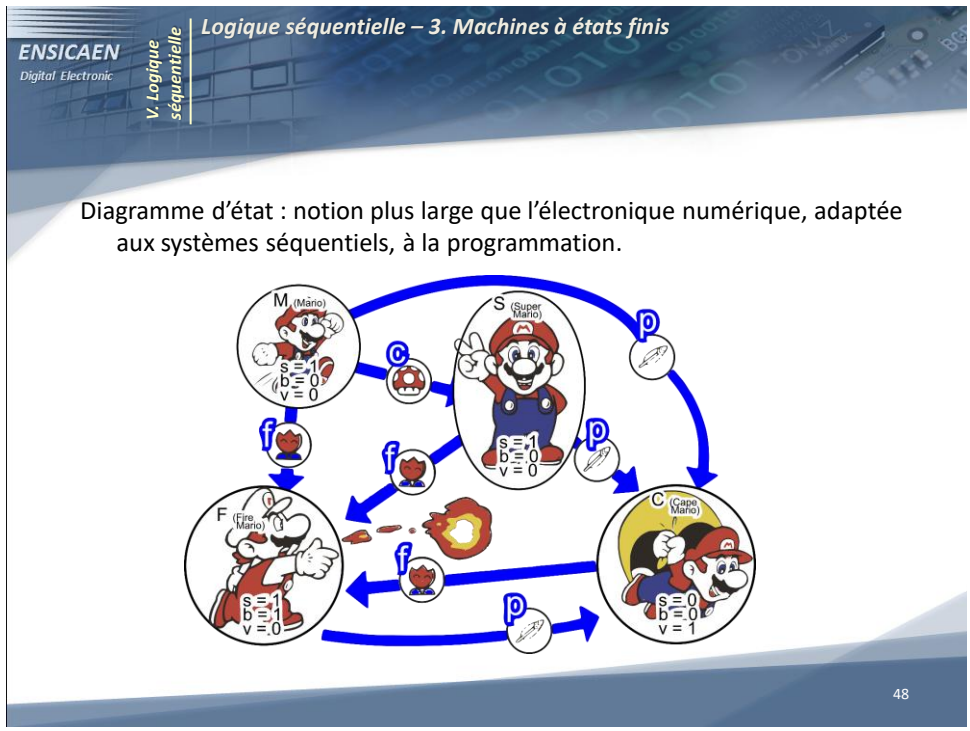
45


ENSICAEN
Digital Electronic
V. Logique séquentielle
Logique séquentielle – 3. Machines à états finis – 3.5. Compteurs (cont.)
Structure (synchrone)
registre d'état
pour générer l'impulsion, ET logique entre les signaux de sortie des bascules
bloc décodage sortie
46

46


ENSICAEN
Digital Electronic
V. Logique séquentielle
Logique séquentielle – 3. Machines à états finis – 3.5. Compteurs (cont.)
Transient Response
/pulse64_async
/pulse64_sync
/clkOut_async
/clkOut_sync
/clk
V (V)
time (us)
structure (synchrone)
structure à éviter (asynchrone)
47

47


ENSICAEN
Digital Electronic
V. Logique séquentielle
Logique séquentielle – 3. Machines à états finis
Diagramme d'état : notion plus large que l'électronique numérique, adaptée aux systèmes séquentiels, à la programmation.
M (Mario)
S (Super Mario)
F (Fire Mario)
C (Cape Mario)
c
p
f
48

48

Dans les CPU/MCU:

Compteurs/décompteurs à préchargement

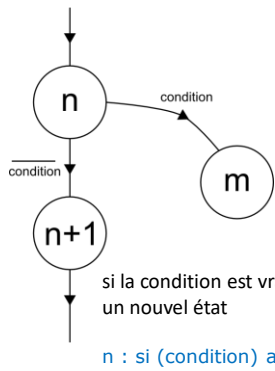
- Compteur programme
- Timer (signal de fin de comptage/décomptage, IT)

Séquenceur

Les machines à états finis sont une des structures pour réaliser des **séquenceurs (automates) dans les cœurs de calculateur**.
L'autre structure est le séquenceur micro-programmé.

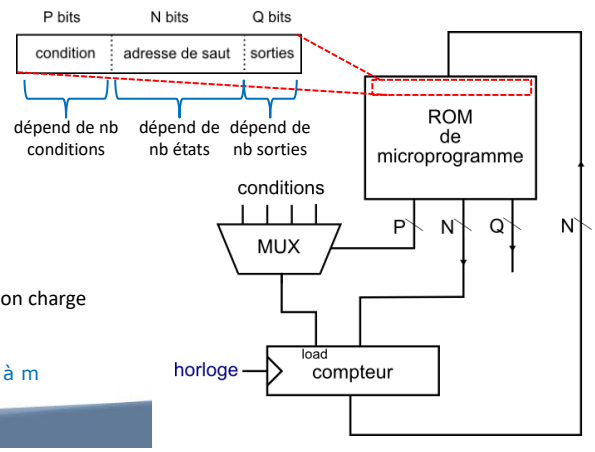
4. Séquenceur micro-programmé

possibilités d'évolution restreintes, moins flexible, moins grande efficacité
branchement conditionnel, incrémentation implicite



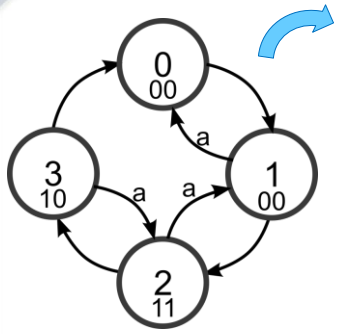
si la condition est vraie, on charge un nouvel état

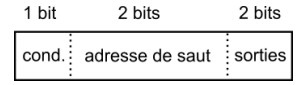
n : si (condition) aller à m
n+1 : si ….



Logique séquentielle – 4. Séquenceur micro-programmé (cont.)

Exemple simple : compteur-décompteur, fonctionnement naturel compteur



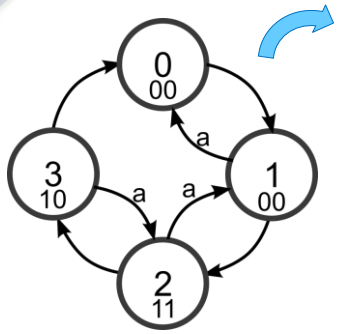


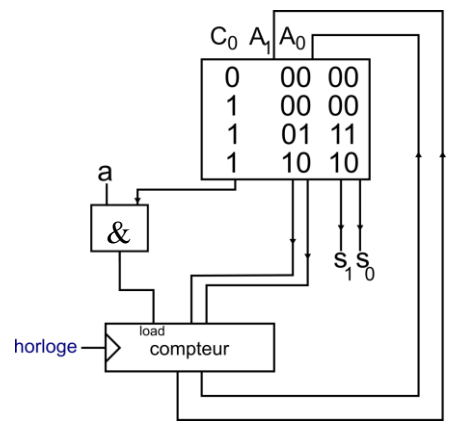
état	Instruction	sorties	micro-instruction		
		S_1S_0	C_0	A_1A_0	S_1S_0
0	----------	00	0	XX	00
1	Si a aller en 0	00	1	00	00
2	Si a aller en 1	11	1	01	11
3	Si a aller en 2	10	1	10	10

a => C0

Logique séquentielle – 4. Séquenceur micro-programmé (cont.)

Exemple : (cont.)

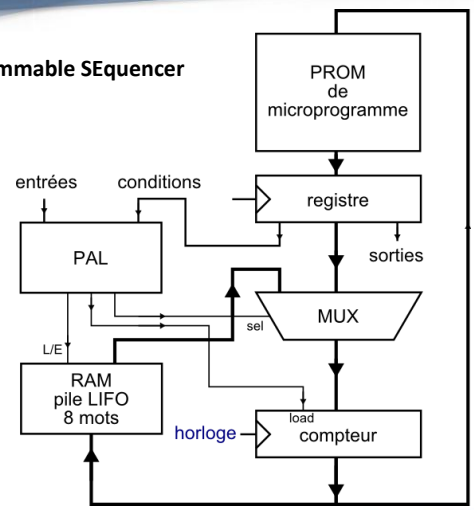




ENSICAEN
Digital Electronic
V. Logique séquentielle

Logique séquentielle – 4. Séquenceur micro-programmé (cont.)

PROSE : PROgrammable SEquencer



53

53

ENSICAEN
Digital Electronic
V. Logique séquentielle

Logique séquentielle – 5. Séquenceur

Séquenceur

Le séquenceur coordonne les sous-opérations nécessaires pour l'exécution des instructions.

<u>solution câblée :</u>
l'automate du séquenceur est une machine à états finis câblée.
privilégiée pour les processeurs RISC (*Reduced Instruction Set Core/Computer*); exemple PIC

<u>solution micro-programmée :</u>
l'automate est un séquenceur micro-programmé (microprogramme contenu dans une ROM)
privilégiée pour les processeurs CISC (*Complex Instruction Set Computer*)

54

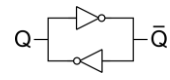
54

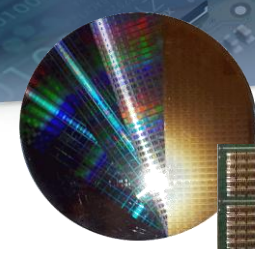
ENSICAEN

ENSICAEN
Digital Electronic
V. Logique séquentielle

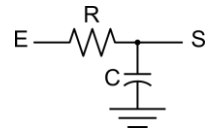
Logique séquentielle. 6. Point mémoire

Point mémoire statique (bistable)



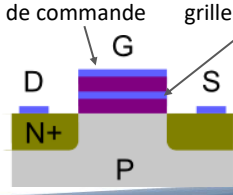


Point mémoire dynamique

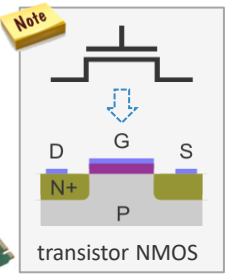


Mémoire flash

grille de commande grille flottante



Note



55

55

ENSICAEN

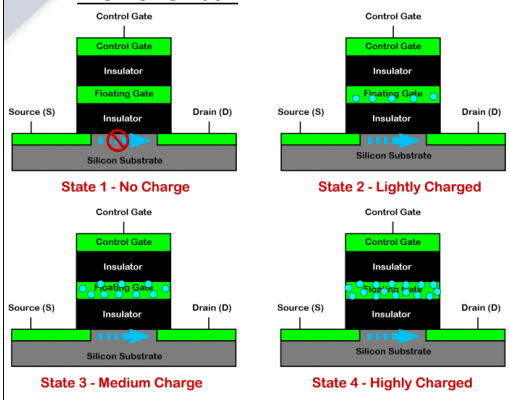
ENSICAEN
Digital Electronic
V. Logique séquentielle

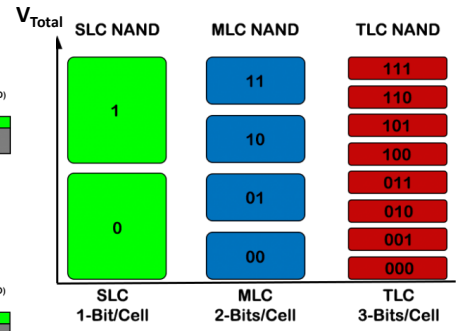
Logique séquentielle. 6. Point mémoire

Inventée par Toshiba en 1980

État de la grille flottante	État logique
Chargée (programmée)	« 0 » - bas
Non chargée (effacée)	« 1 » - haut

Mémoire flash





https://www.cactus-tech.com/resources/blog/details/solid-state-drive-primer-2-slc-mlc-and-tlc-nand-flash/

MLC (Multi Level Cell) TLC (Tri Level Cell)

56

56

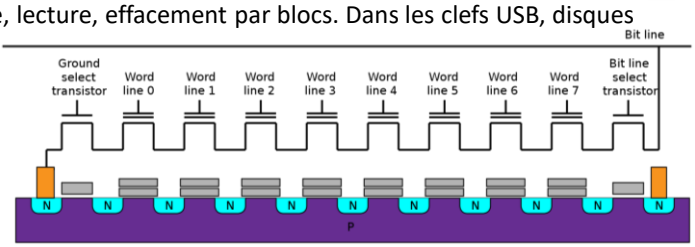
ENSICAEN Digital Electronic | V. Logique séquentielle

Logique séquentielle. 6. Point mémoire

Mémoire flash

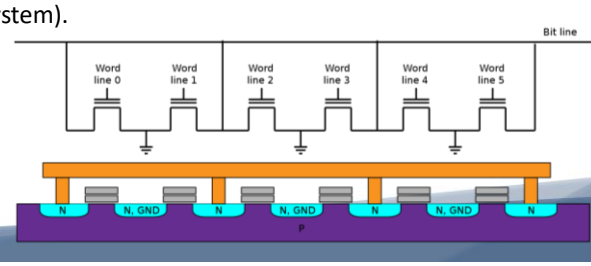
Flash NAND : écriture, lecture, effacement par blocs. Dans les clefs USB, disques durs, smartphones.

Transistors en série. Bit line à «0» quand toutes les Word lines à «1»



Flash NOR : écriture, lecture, effacement par mots. Stockage du code exécutable dans les microcontrôleurs, mémoire de démarrage (Boot ROM), firmware et BIOS (Basic Input/Ouput System).

«1» sur Word line programme un «0»

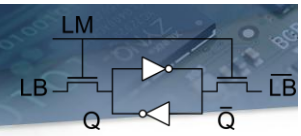


57

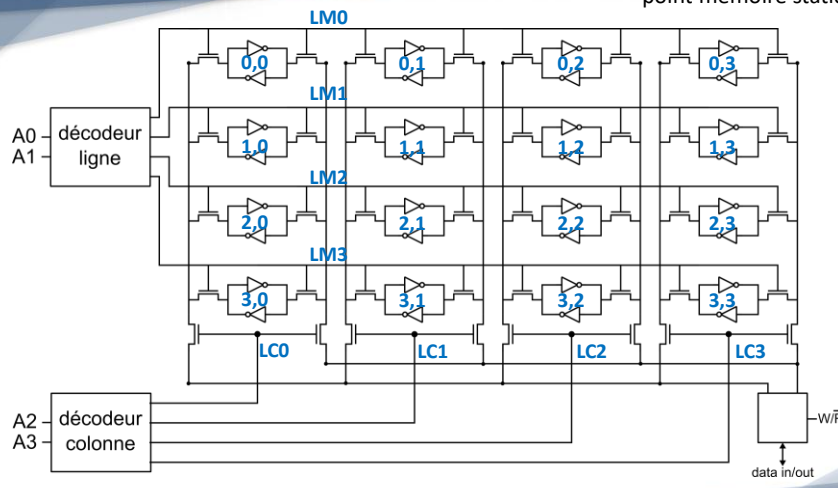
57

ENSICAEN Digital Electronic | V. Logique séquentielle

Logique séquentielle. 6. Point mémoire (cont.)



point mémoire statique



16 mots de 1 bit

58

58

ENSICAEN

ENSICAEN
Digital Electronic
V. Logique séquentielle

A l'issue de cette cinquième partie (1/2)

Quelle différence essentielle existe-t-il entre un système asynchrone et un système synchrone ?

Quels sont les avantages/inconvénients respectifs des systèmes asynchrones et synchrones ?

De quel type sont la grande majorité des microprocesseurs et microcontrôleurs (synchrone ou asynchrone) ?

59

59

ENSICAEN
Digital Electronic
V. Logique séquentielle

A l'issue de cette cinquième partie (2/2)

Quels sont les deux types de machine à états finis ?

Quelles sont les étapes de la synthèse d'une machine à états finis ?

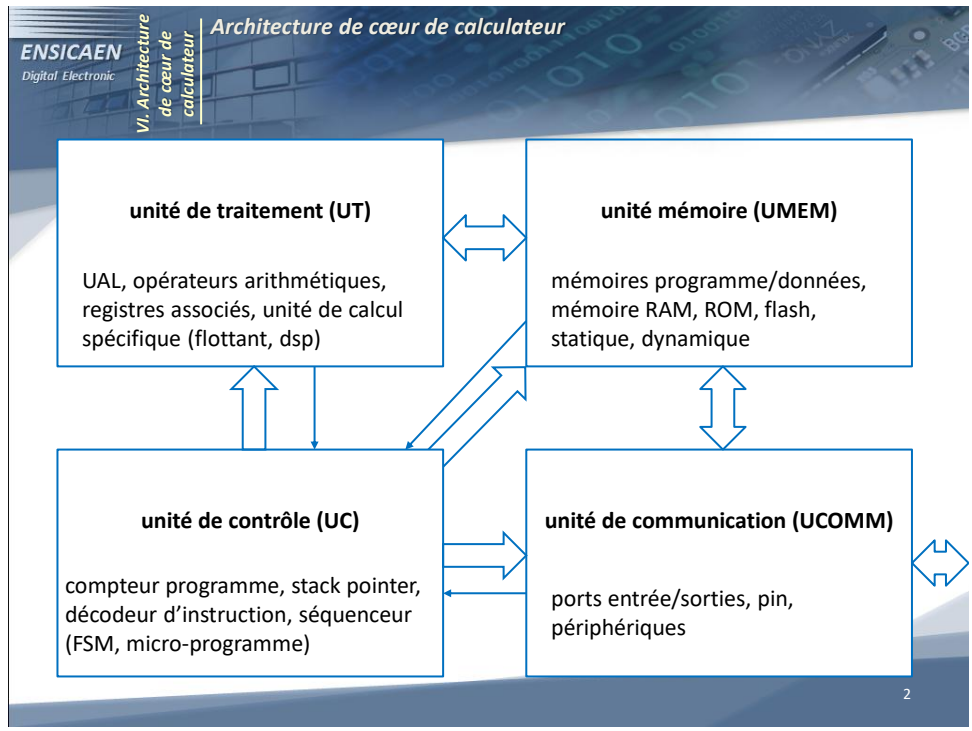
Quels sont les avantages/inconvénients relatifs des solutions machine à états finis et séquenceur microprogrammé pour la synthèse de circuits logiques séquentiels ?

60

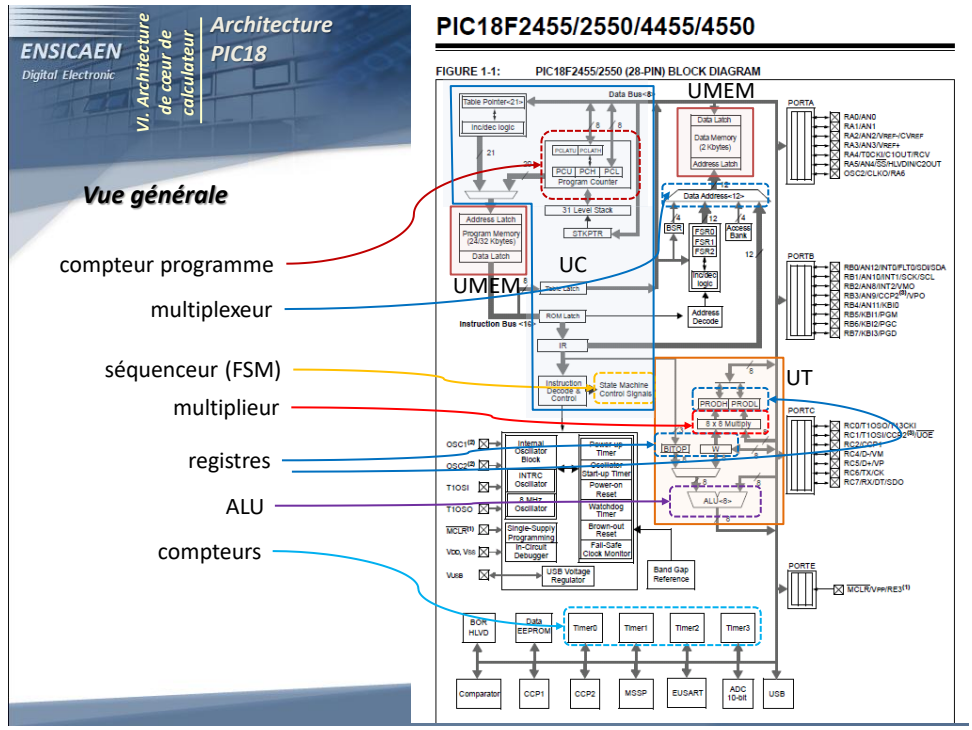
60


ENSICAEN
Digital Electronic
VI. Architecture de cœur de calculateur
UT (UAL), UC, Umem, Ucomm
M. Denoual
1

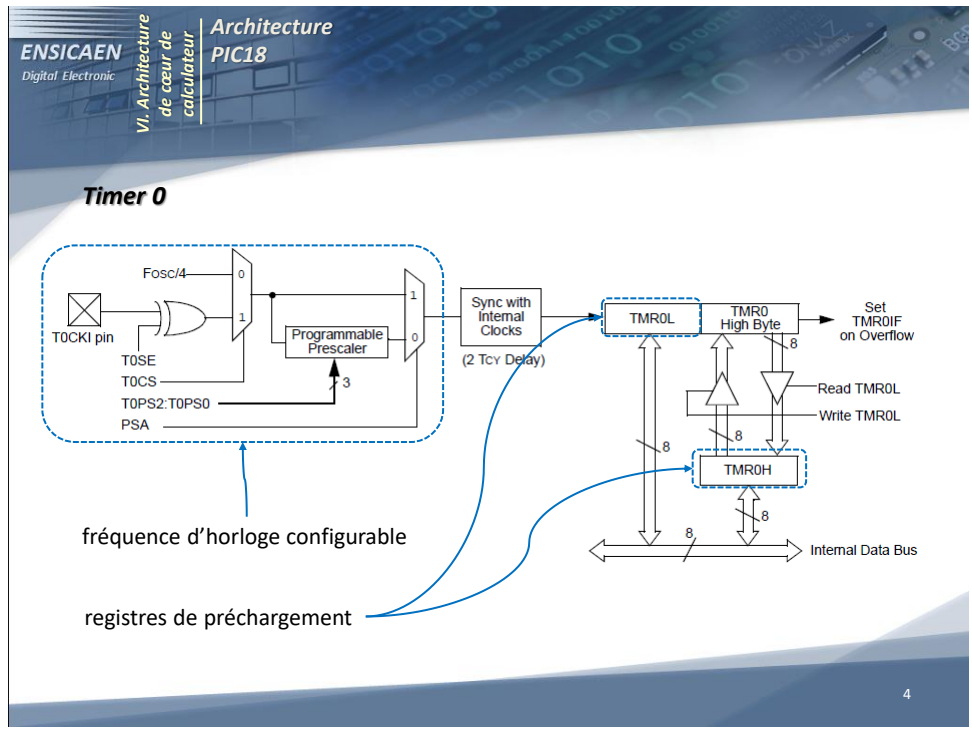
1


ENSICAEN
Digital Electronic
VI. Architecture de cœur de calculateur
Architecture de cœur de calculateur
unité de traitement (UT)
UAL, opérateurs arithmétiques, registres associés, unité de calcul spécifique (flottant, dsp)
unité mémoire (UMEM)
mémoires programme/données, mémoire RAM, ROM, flash, statique, dynamique
unité de contrôle (UC)
compteur programme, stack pointer, décodeur d'instruction, séquenceur (FSM, micro-programme)
unité de communication (UCOMM)
ports entrée/sorties, pin, périphériques
2

2


ENSICAEN
Digital Electronic
VI. Architecture de cœur de calculateur
Architecture PIC18
Vue générale
compteur programme
multiplexeur
séquenceur (FSM)
multiplieur
registres
ALU
compteurs
PIC18F2455/2550/4455/4550
FIGURE 1-1: PIC18F2455/2550 (28-PIN) BLOCK DIAGRAM
UMEM
UC
UT
PORTA
PORTB
PORTC
PORTE
Timer0
Timer1
Timer2
Timer3
Comparator
CCP1
CCP2
MSSP
EUSART
ADC 10-bit
USB

3


ENSICAEN
Digital Electronic
VI. Architecture de cœur de calculateur
Architecture PIC18
Timer 0
Fosc/4
T0CKI pin
T0SE
T0CS
T0PS2:T0PS0
PSA
Programmable Prescaler
Sync with Internal Clocks
(2 TCY Delay)
TMR0L
TMR0 High Byte
Set TMR0IF on Overflow
Read TMR0L
Write TMR0L
TMR0H
Internal Data Bus
fréquence d'horloge configurable
registres de préchargement
4

4

ENSICAEN
Digital Electronic

VI. Architecture de cœur de calculateur

Architecture PIC18

Multiplication

PIC18F2455/2550/4455/4550

8.0 8 x 8 HARDWARE MULTIPLIER

8.1 Introduction

All PIC18 devices include an 8 x 8 hardware multiplier as part of the ALU. The multiplier performs an unsigned operation and yields a 16-bit result that is stored in the product register pair, PRODH:PRODL. The multiplier's operation does not affect any flags in the Status register.

Making multiplication a hardware operation allows it to be completed in a single instruction cycle. This has the advantages of higher computational throughput and reduced code size for multiplication algorithms and allows the PIC18 devices to be used in many applications previously reserved for digital signal processors. A comparison of various hardware and software multiply operations, along with the savings in memory and execution time, is shown in Table 8-1.

8.2 Operation

Example 8-1 shows the instruction sequence for an 8 x 8 unsigned multiplication. Only one instruction is required when one of the arguments is already loaded in the WREG register.

Example 8-2 shows the sequence to do an 8 x 8 signed multiplication. To account for the sign bits of the arguments, each argument's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EXAMPLE 8-1: 8 x 8 UNSIGNED MULTIPLY ROUTINE

```
MOVF   ARG1, W   ;
MULWF  ARG2      ; ARG1 * ARG2 ->
                 ; PRODH:PRODL
```

EXAMPLE 8-2: 8 x 8 SIGNED MULTIPLY ROUTINE

```
MOVF   ARG1, W
MULWF  ARG2      ; ARG1 * ARG2 ->
                 ; PRODH:PRODL
BTFSC  ARG2, SB  ; Test Sign Bit
SUBWF  PRODH, F  ; PRODH = PRODH
                 ;       - ARG1
MOVF   ARG2, W
BTFSC  ARG1, SB  ; Test Sign Bit
SUBWF  PRODH, F  ; PRODH = PRODH
                 ;       - ARG2
```

TABLE 8-1: PERFORMANCE COMPARISON FOR VARIOUS MULTIPLY OPERATIONS

Routine	Multiply Method	Program Memory (Words)	Cycles (Max)	Time @ 40 MHz	Time @ 10 MHz	Time @ 4 MHz
8 x 8 unsigned	Without hardware multiply	13	69	6.9 µs	27.6 µs	69 µs
	Hardware multiply	1	1	100 ns	400 ns	1 µs
8 x 8 signed	Without hardware multiply	33	91	9.1 µs	36.4 µs	91 µs
	Hardware multiply	6	6	600 ns	2.4 µs	6 µs
16 x 16 unsigned	Without hardware multiply	21	242	24.2 µs	96.8 µs	242 µs
	Hardware multiply	28	28	2.8 µs	11.2 µs	28 µs
16 x 16 signed	Without hardware multiply	52	254	25.4 µs	102.6 µs	254 µs
	Hardware multiply	35	40	4.0 µs	16.0 µs	40 µs

5

ENSICAEN
Digital Electronic

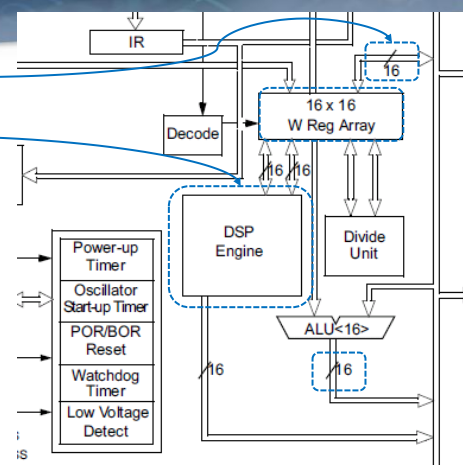
VI. Architecture de cœur de calculateur

Architecture dsPIC30

Cœur

A DSP engine has been included to significantly enhance the core arithmetic capability and throughput. It features a high speed 17-bit by 17-bit multiplier, a 40-bit ALU, two 40-bit saturating accumulators and a 40-bit bidirectional barrel shifter. Data in the accumulator or any working register can be shifted up to 15 bits right, or 16 bits left in a single cycle. The DSP instructions operate seamlessly with all other instructions and have been designed for optimal real-time performance. The MAC class of instructions can concurrently fetch two data operands from memory while multiplying two W registers. To enable this concurrent fetching of data operands, the data space has been split for these instructions and linear for all others. This has been achieved in a transparent and flexible manner, by dedicating certain working registers to each address space for the MAC class of instructions.

16 bits

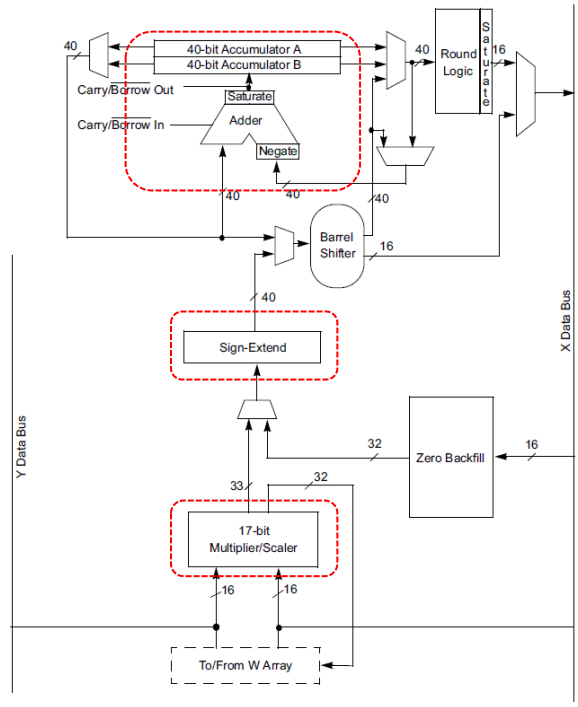


6

6

ENSICAEN
Digital Electronic
VI. Architecture de cœur de calculateur
Architecture dsPIC

DSP engine



7

ENSICAEN
Digital Electronic
VI. Architecture de cœur de calculateur
Architecture dsPIC30

Multiplication

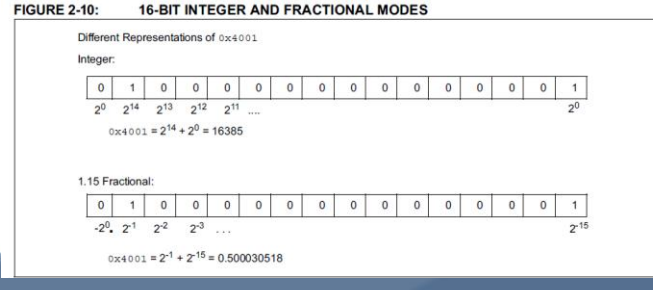
dsPIC30F

2.5.1 MULTIPLIER

The 17 x 17-bit multiplier is capable of signed or unsigned operation and can multiplex its output using a scaler to support either 1.31 fractional (Q31) or 32-bit integer results. The respective number representation formats are shown in Figure 2-10. Unsigned operands are zero-extended into the 17th bit of the multiplier input value. Signed operands are sign-extended into the 17th bit of the multiplier input value. The output of the 17 x 17-bit multiplier/scaler is a 33-bit value which is sign-extended to 40 bits. Integer data is inherently represented as a signed two's complement value, where the MSB is defined as a sign bit. Generally speaking, the range of an N-bit two's complement integer is -2^{N-1} to $2^{N-1} - 1$. For a 16-bit integer, the data range is -32768 (0x8000) to 32767 (0x7FFF) including '0' (see Figure 2-10). For a 32-bit integer, the data range is -2,147,483,648 (0x8000 0000) to 2,147,483,645 (0x7FFF FFFF).

When the multiplier is configured for fractional multiplication, the data is represented as a two's complement fraction, where the MSB is defined as a sign bit and the radix point is implied to lie just after the sign bit (QX format). The range of an N-bit two's complement fraction with this implied radix point is -1.0 to $(1 - 2^{1-N})$. For a 16-bit fraction, the Q15 data range is -1.0 (0x8000) to 0.999969482 (0x7FFF) including '0' and has a precision of 3.01518×10^{-5}. In Fractional mode, the 16x16 multiply operation generates a 1.31 product which has a precision of 4.65661×10^{-10}.

FIGURE 2-10: 16-BIT INTEGER AND FRACTIONAL MODES



8

ENSICAEN Digital Electronic | VI. Architecture de cœur de calculateur | Architecture dsPIC30

Multiplication

Assembly Syntax		Description	# of Words	# of Cycles	Status Flags Affected
MPY	Wm*Wn,Acc,Wx,Wxd,Wy,Wyd	Multiply Wm by Wn to Accumulator	1	1	OA,OB,OAB, SA,SB,SAB
MPY	Wm*Wm,Acc,Wx,Wxd,Wy,Wyd	Square Wm to Accumulator	1	1	OA,OB,OAB, SA,SB,SAB
MPY.N	Wm*Wn,Acc,Wx,Wxd,Wy,Wyd	-(Multiply Wm by Wn) to Accumulator	1	1	None
MSC	Wm*Wm,Acc,Wx,Wxd,Wy,Wyd, AWB	Multiply and Subtract from Accumulator	1	1	OA,OB,OAB, SA,SB,SAB
MUL.SS	Wb,Ws,Wnd	{Wnd+1, Wnd} = signed(Wb) * signed(Ws)	1	1	None
MUL.SU	Wb,Ws,Wnd	{Wnd+1, Wnd} = signed(Wb) * unsigned(Ws)	1	1	None
MUL.US	Wb,Ws,Wnd	{Wnd+1, Wnd} = unsigned(Wb) * signed(Ws)	1	1	None
MUL.UU	Wb,Ws,Wnd	{Wnd+1, Wnd} = unsigned(Wb) * unsigned(Ws)	1	1	None
MUL.SU	Wb,#lit5,Wnd	{Wnd+1, Wnd} = signed(Wb) * unsigned(lit5)	1	1	None
MUL.UU	Wb,#lit5,Wnd	{Wnd+1, Wnd} = unsigned(Wb) * unsigned(lit5)	1	1	None
MUL	f	W3:W2 = f * WREG	1	1	None

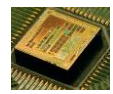
9

9

ENSICAEN Digital Electronic | VI. Architecture de cœur de calculateur | Conclusion, électronique numérique

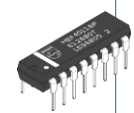
solutions spécifiques/standards

circuit intégré dédié
ASIC : Application Specific Integrated Circuit
SoC : System-on-Chip
Sip : System-in-Package

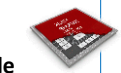


architecture à CPU
microprocesseur
microcontrôleur
DSP

composants discrets
très petites applications

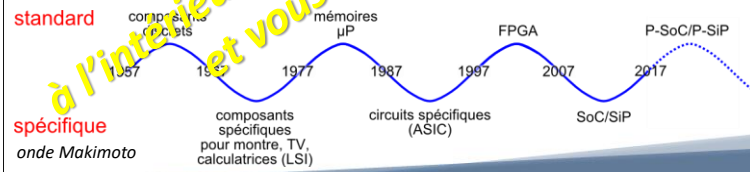


composants à logique programmable
PLD, FPGA



System-on-chip programmable
PSoC

à l'intérieur de tout cela, il y a des circuits logiques et vous avez vu comment c'est fait !



10

10