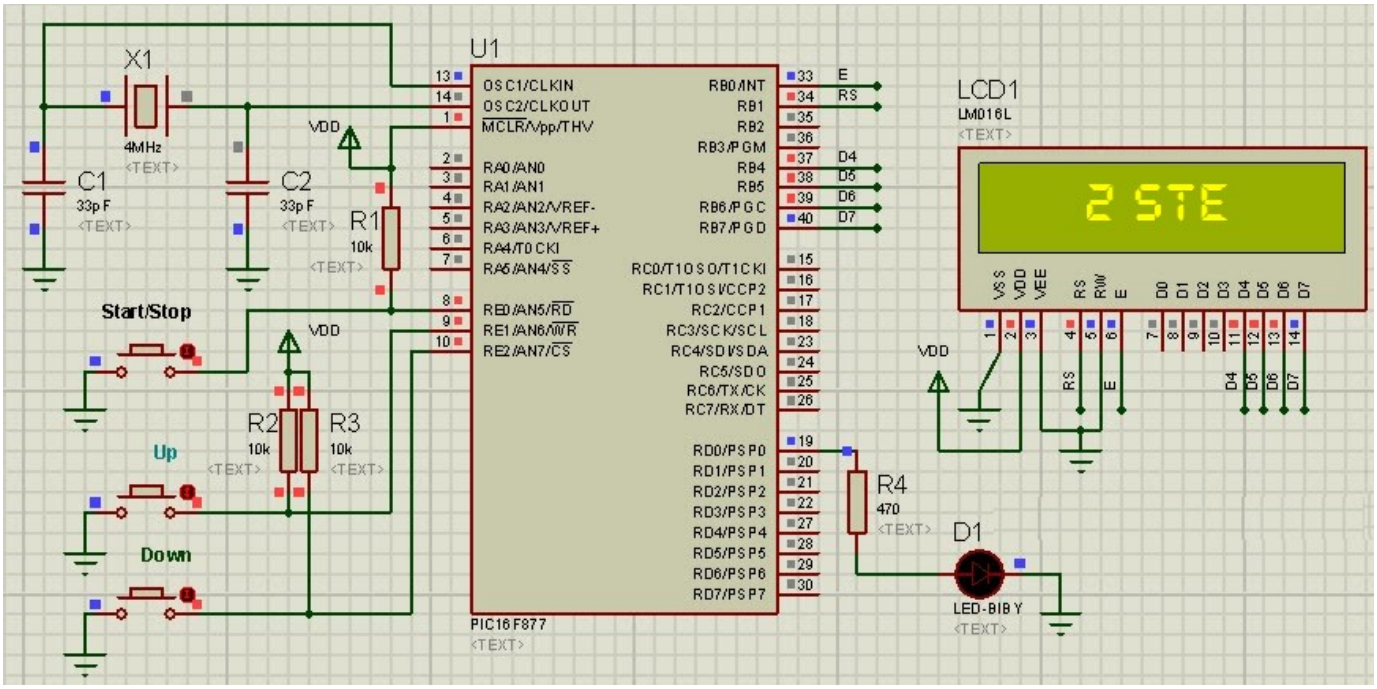
PARTIE

2

La fonction TRAITER



CIRCUITS LOGIQUES PROGRAMMABLES

Un circuit logique programmable PLD (Programmable Logic Device) est un composant logique disposant d'entrées et de sorties dont on peut programmer le schéma selon la fonction souhaitée, combinatoire et/ou séquentielle

Il apporte un gain dans la mesure où :

- Un seul circuit peut remplacer plusieurs
- La taille et le coût de la fonction implantée sont moindres

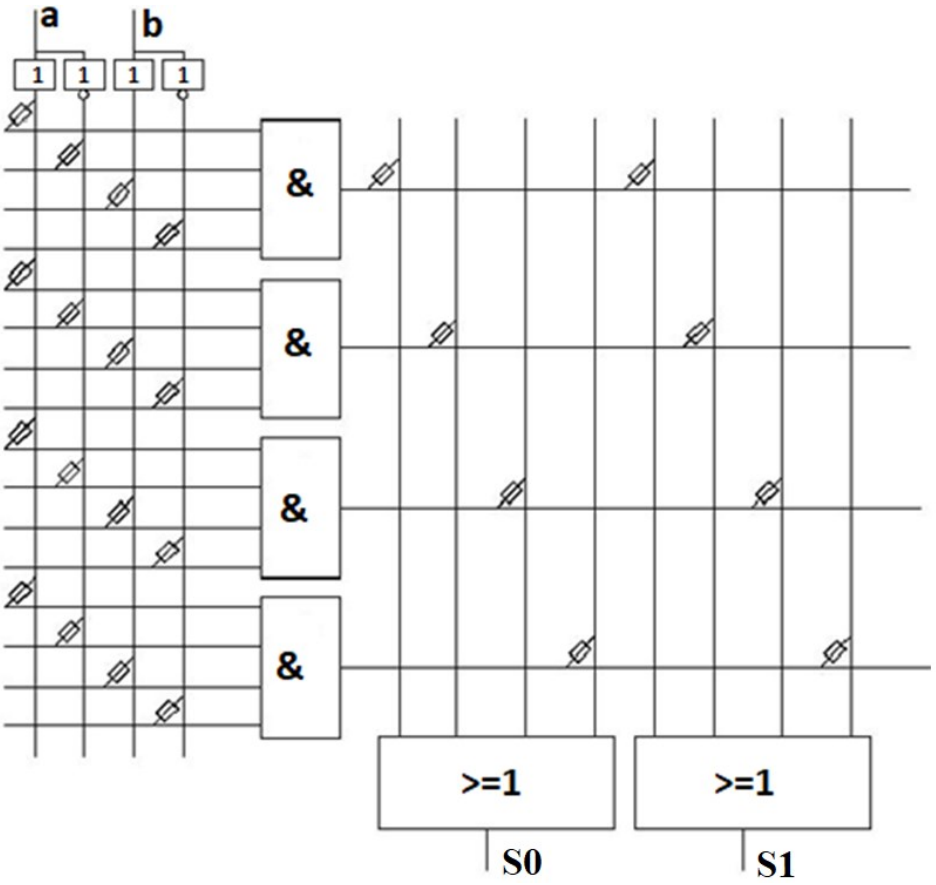
Architecture d'un PLD

Toute fonction logique peut être réalisée par seules les portes logiques fondamentales ET, OU et NON
Un PLD est bâti autour de ce concept

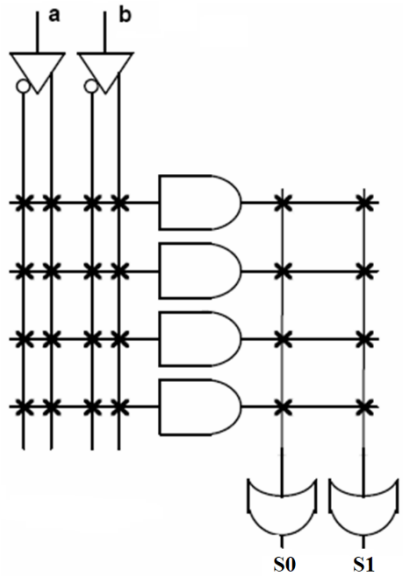
La plupart des PLD adoptent la structure suivante :

- Une matrice d'opérateurs ET sur lesquels viennent se connecter les variables d'entrée
- Une matrice d'opérateurs OU sur lesquels les sorties des opérateurs ET sont connectées.
- Une éventuelle structure de sortie (Portes inverseuses, bascules...).

Cas d'un PLD à 2 entrées et 2 sorties combinatoires (structure très simplifiée, avant la programmation)

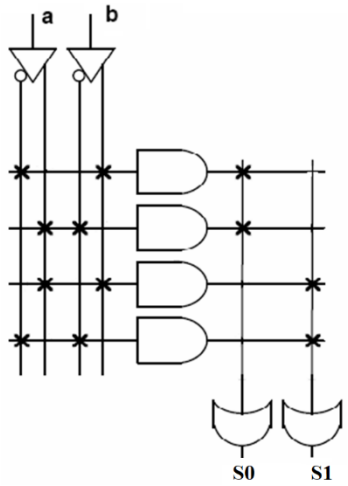


Représentation simplifiée



Les fonctions programmées ci-contre sont :

$Q0 = \overline{a}b + a\overline{b}$
$Q1 = ab + \overline{a}\overline{b}$



Exercice

Générateur de parité à 3 entrées

S = 1 si le nombre de 1 à l'entrée est impair;
S = 0 sinon

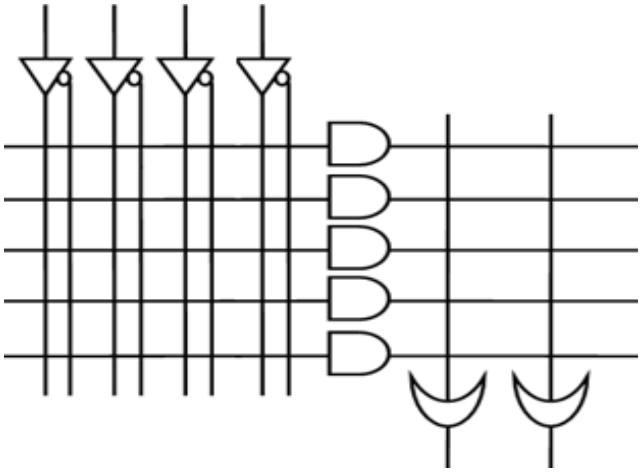
Table de vérité

a	b	c	S
0	0	0	……
0	0	1	……
0	1	0	……
0	1	1	……
1	0	0	……
1	0	1	……
1	1	0	……
1	1	1	……

Equation

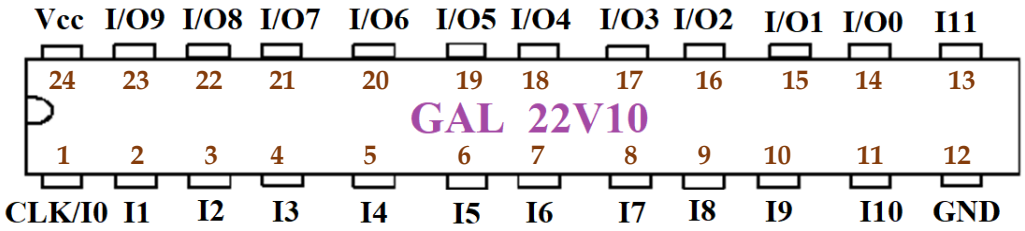
S = ……

Programmation du PLD



Le GAL 22V10

Avec les évolutions techniques actuelles, les PLD sont de plus en plus rapides, à plus grand nombre d'E/S, effaçables et reprogrammables; on trouve alors différents types de PLD : PAL, GAL, CPLD, FPGA…



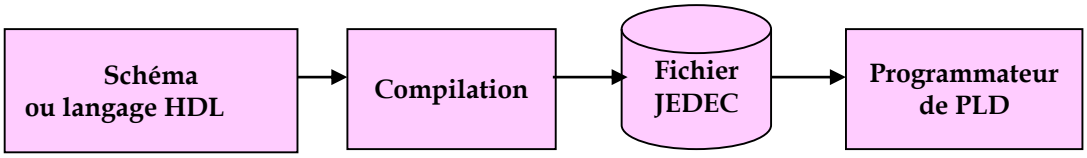
La désignation 22V10 signifie que le composant dispose de 22 entrées possibles et 10 sorties possibles
Chaque broche output émane d'une cellule OLMC (Output Logic MacroCell) qui définit par programme la configuration de la sortie : combinatoire ou séquentielle, normale ou complémentaire

Programmation des PLD: langage Abel

La programmation des PLD nécessite un programmateur, un microordinateur et un logiciel adapté
Souvent le logiciel offre la possibilité de décrire la fonction à réaliser :

- Soit par schéma
- Soit syntaxiquement moyennant un langage de programmation HDL : ABEL, VHDL ou VERILOG

Lorsqu'il est terminé, le fichier de description (schéma ou programme) doit être compilé pour être traduit en format JEDEC, c'est un fichier contenant des 0 et des 1 signifiant les croisements de fils qui seront détruits et les autres qui seront maintenus. C'est ce fichier JEDEC qui est transféré au PLD via le programmateur



Voici un exemple de programme Abel et le Jedec correspondant

```
module test
Declarations
      test  device 'P22V10';
      a,b   pin 2,3;
      s1,s2  pin 14,15 istype 'com';
Equations
      s1    = a # b;
      s2    = !a;
end test
```

```
QP24* QF5828* QV0* F0*
 X0*
NOTE Table of pin names and numbers*
NOTE PINS a:2 b:3 s1:14 s2:15*
L4884 11111111111111111111111111111111111111111111*
L4928 11110111011111111111111111111111111111111111*
L4972 11110111011111111111111111111111111111111111*
L5368 11111111111111111111111111111111111111111111*
L5412 11110111011111111111111111111111111111111111*
L5456 11110111101111111111111111111111111111111111*
L5808 00000000000000001111*
C2097*
```

⇨ *Structure d'un programme ABEL*

On y trouve :

- La section d'entête → MODULE, TITLE
- La section des déclarations → DECLARATIONS (DEVICE, E/S, bus…)
- La section de description → EQUATIONS, TRUTH_TABLE, STATE_DIAGRAM
- La déclaration de fin du fichier → END

⇨ *Les déclarations*

- **Pin :** permet d'affecter une broche d'E/S à une variable
  Exemples

```
e1, e2, A1 pin 2, 3, 4;        : entrées affectées aux broches 2 ,3 et 4
d4 pin ;                       : entrée sans désignation de numéro de broche
s1  pin 12 istype 'com';       : sortie combinatoire associée à la broche 12
S0..S6 pin istype 'com';       : sorties combinatoires
S1 pin  istype 'reg';          : sortie séquentielle
```

- On peut déclarer des ensembles (bus) de variables
  Exemples N = [D2, D1, D0] ; A = [b7..b0] ;
  L'affectation N=5 entraîne D2=1 , D1 = 0 et D0 = 1

⇨ *Les opérateurs*

Opérateurs logiques

```
! :  NON
& :  ET
# :  OU
$ :  OU exclusif
!$ : NON OU exclusif
```

Opérateurs relationnels

```
==   Egal
!=   Différent
<    Inférieur
<=   Inférieur ou égal
>    Supérieur
>=   Supérieur ou égal
```

Opérateurs arithmétiques

```
-   négation ou soustraction
+   Addition
*   Multiplication
/   Division
%   Reste de la division
```

⇨ *Instructions d'affectation et de test*

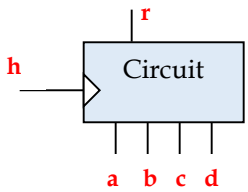
- S = a & !b # !a & b; — affectation pour une sortie combinatoire (symbole **=**)
  A0 **:=** !D1 & A0 & !D7; — affectation pour une sortie séquentielle (symbole **:=**)
  X = a<b — X reçoit 1 si a<b et X reçoit 0 sinon

- La structure conditionnelle — **WHEN** (condition) **THEN** équation1 ; **ELSE** équation2 ;
  Exemple — when (E1 == 0) then S = a $ b ; else S = a !$ b ;

⇨ *Les extensions*

Les pins peuvent être suivis d'une extension leur associant une fonction particulière

- **.CLK** : Entrée horloge
- **.AR** : Reset asynchrone
- **.OE** : Output Enable
- **.D** : Entrée D d'une bascule D
- **.J** : Entrée J d'une bascule JK
- **.K** : Entrée K d'une bascule JK



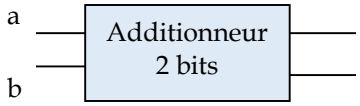
Dans cet exemple, on déclare que

- L'entrée **h** est l'horloge des bascules de sortie
- L'entrée **r** effectue une mise à 0 asynchrone des sorties

```
..
Declarations
   h, r   pin  ;
   a, b, c, d   pin  istype 'reg';
   S =[a, b, c, d];
Equations
   S.CLK = h ;
   S.AR = r ;
..
```

Exercices

Additionneur 2 bits



a	b	S	R
0	0	……	……
0	1	……	……
1	0	……	……
1	1	……	……

S = ……..…..…..

R = ……..…..…..

Description par la table de vérité

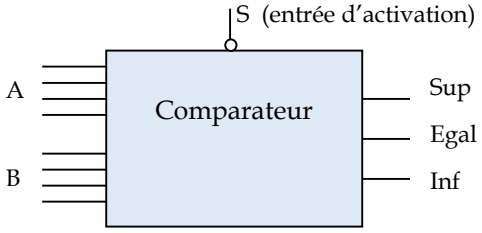
```
MODULE additionneur
DECLARATIONS
      additionneur DEVICE 'P22V10' ;
      a, b PIN ;
      S, R  PIN  ISTYPE 'COM' ;
TRUTH_TABLE ( [ ……..…] ->[…….....] )
      ……
      ……
      ……
      ……
END additionneur
```

Description par les équations logiques

```
MODULE additionneur
DECLARATIONS
   additionneur DEVICE 'P22V10' ;
   a, b PIN  4, 5 ;
   S, R  PIN  15, 16 ISTYPE 'COM' ;
EQUATIONS
   S =  ……
   R = ……
END additionneur
```

Comparateur 4 bits

Sup = 1 si A>B
Inf = 1 si A<B
Egal = 1 si A=B



module comparateur
Declarations

……
……
……
……
……

Equations

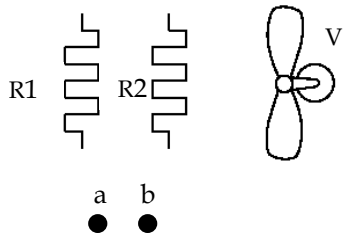
……
……
……
……

End ……

Chauffage électrique

Le système de chauffage est équipé d'un radiateur électrique à deux allures de chauffe, comportant deux résistances R1 et R2 et un ventilateur V. Il est commandé par deux interrupteurs a et b.

- Par action sur a seul, la résistance R1 seule est mise sous-tension.
- Par action sur b seul ou sur a et b à la fois, les deux résistances et le ventilateur sont mis sous-tension.



module ……
Declarations

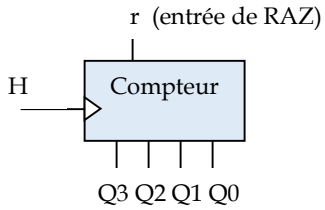
……
……
……

Truth_Table (……)

……
……
……
……

End ……

Compteur 4 bits avec mise à 0 asynchrone

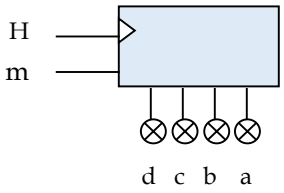


module ……..
Declarations
……
……
……
……
equations
……
……
……

End ……..

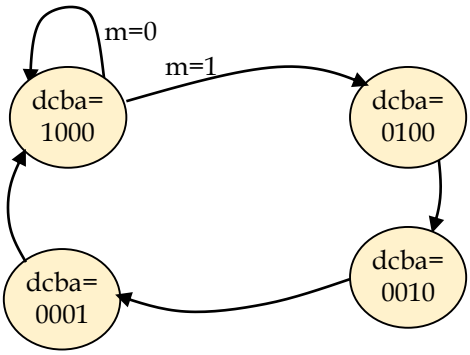
Chenillard à 4 LED

Au départ la Led **d** est allumé. L'appui sur le bouton **m** lance une séquences de défilements vers la droite



Le diagramme d'état est une représentation symbolique de l'enchaînement séquentiel des états de sortie d'un système.

Le diagramme d'état suivant illustre le cycle d'allumage désiré



Module chenillard
DECLARATIONS
chenillard device 'P22V10' ;
……..
……
……
EQUATIONS
……
STATE_DIAGRAM ……
State [1,0,0,0] : IF (m==1) THEN goto [0,1,0,0] ; else goto [1,0,0,0] ;
……
……
……

END chenillard