GENERATION D'IMAGES VIDEO PAR CIRCUIT LOGIQUE PROGRAMMABLE

INTRODUCTION

Le présent document contient les éléments techniques et pédagogiques permettant de bâtir une activité relative au domaine de la vidéo avec des élèves de classe de Technicien Supérieur en Electronique.

Le domaine de la vidéo est une partie de l'Electronique qui ne peut pas être ignorée en section de BTS (Voir le référentiel : *DISPOSITIFS DU DOMAINE GRAND PUBLIC - Dispositifs de restitution du son et de l'image - Dispositifs de mémorisation du son et de l'image*).
Bien souvent le travail effectué sur ce thème s'appuie sur un cours de type "magistral" étayé par des expérimentations constituées le plus souvent par des analyses de signaux existants. Ces signaux étant complexes, ils nécessitent, pour une analyse assez fine, des matériels spécifiques et généralement indisponibles dans les établissements.
Il ne sera généralement pas possible, avec les oscilloscopes qui équipent couramment les laboratoires, d'analyser le contenu de chaque ligne et de faire la liaison entre ce contenu et l'image obtenue.

L'objectif des activités proposées dans ce document est, justement, de permettre une plus grande maîtrise par les élèves des liens existants entre le contenu d'un signal vidéo et l'image qu'il forme sur un écran.
De plus, les moyens qui devront être utilisés les conduiront à améliorer leur pratique de la mise en oeuvre de circuits logiques programmables.

Effectivement, ces activités consistent en des analyses et synthèses qui concourent à la conception d'un montage électronique. Celui-ci est conçu pour générer les différents signaux nécessaires à la visualisation d'images simples sur l'écran d'un moniteur vidéo.
Ces différents signaux, avant leur mise en conformité avec le standard de liaison au moniteur, sont fournis par un sous-ensemble logique. La mise en oeuvre d'un tel sous-ensemble, utilise avantageusement un circuit logique programmable. Dans ce cas, le montage complet se réduit à très peu de composants et peut facilement être réalisé. De plus, la reprogrammation aisée des circuits logiques permet d'envisager différentes expérimentations sans perte de temps dans les opérations de câblage.

Le présent document est organisé de la manière suivante :
- Une première partie fait l'inventaire des prérequis nécessaires.
- La seconde partie est constituée par un exemple d'énoncé de travail demandé aux élèves.
- La troisième partie donne des éléments de réponse à ce travail.
- La quatrième partie propose le texte d'une évaluation individuelle écrite en temps limité.
- Dans la dernière partie, on envisage l'aspect matériel de ce travail quant **aux choix du circuit logique programmable** et du moniteur.

L'exemple de travail dont l'énoncé est donné dans la seconde partie, utilise un circuit IspLSI-1016 de LATTICE. Il s'agit d'un circuit logique programmable de type CPLD (*Complex Programmable Logic Device*) dont la structure interne est dérivée de celle des GAL (*Generic Array Logic*).
Ce circuit possède des ressources bien plus importantes qu'un circuit logique programmable standard comme le GAL22V10. En effet, celui-ci comprend 12 entrées et 10 sorties pouvant être réutilisées comme entrées (10 registres de type D au total) alors que le circuit IspLSI-1016 est composé de 36 entrées-sorties et de 64 registres de type D ou T.

En annexe, on trouvera une classification simplifiée des circuits logiques programmables.

PRE-REQUIS

Afin que les élèves puissent aborder ce projet avec suffisamment d'autonomie, ils doivent être déjà familiers avec certains domaines.

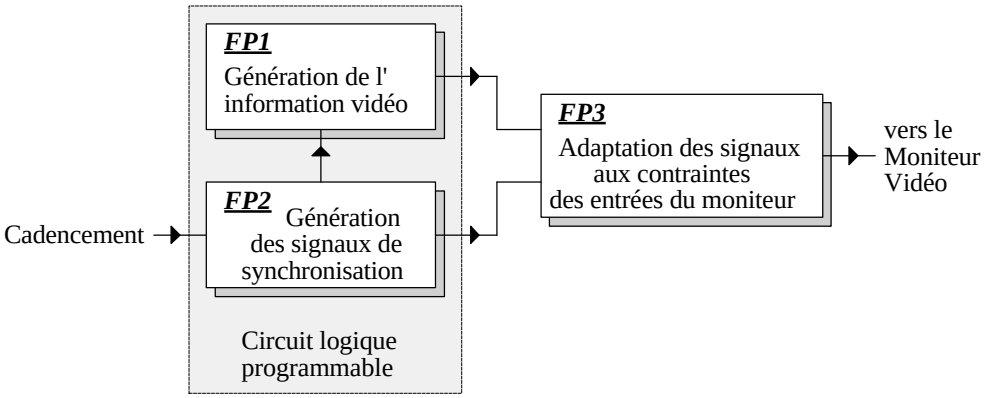
- Domaine de la vidéo :
  - Formation d'une image sur un écran cathodique :
    - * le tube cathodique monochrome et couleur,
    - * balayages horizontaux et verticaux, simples et entrelaçés.
  - Organisation fonctionnelle d'un moniteur :
    - * circuits de balayages horizontaux et verticaux,
    - * les amplificateurs vidéo.
  - Définition des signaux à fournir à un moniteur :
    - * signaux de synchronisations horizontale et verticale (ligne et trame),
    - * signaux vidéo (monochrome et couleur),
    - * signaux synchro-composite et vidéo-composite.
  - Définition des différents standards en matière de moniteur vidéo :
    - * partie moniteur d'un poste de télévision (après la prise Péritel),
    - * moniteur informatique (MDA, CGA, Hercule, EGA, VGA, etc..).

- Domaine de la logique :
  - Fonctions logiques combinatoires et séquentielles de base et plus particulièrement :
    - * fonctionnement asynchrone et synchrone,
    - * comptage binaire asynchrone et synchrone modulo N,
    - * attributs de la fonction comptage (Remise à 0, prépositionnement, décomptage, etc..).
  - Descriptions des fonctions logiques de base en langage comportemental (ABEL ou VHDL).

- Mise en oeuvre de circuits logiques programmables :
  - Les circuits logiques programmables :
    - * classification par type (ASIC, PLD, CPLD, FPGA),
    - * architecture interne :
      - = organisation (nombre et dimension des cellules, type d'interconnection entre cellules, etc..),
      - = type de ressource combinatoire (Mémoire ou *Look Up Table*, PAL ou PLA),
    - * technologie :
      - = technologie utilisée pour mémoriser la structure programmée,
      - = technologie des cellules logiques avec leurs caractéristiques principales.
  - Mise en oeuvre du circuit logique programmable utilisé :
    - * description des différentes étapes du processus,
    - * utilisation de l'outil de développement,
    - * utilisation de l'outil de programmation du composant,
    - * description et caractéristiques technologiques du composant utilisé.

ENONCE DU TRAVAIL A EFFECTUER

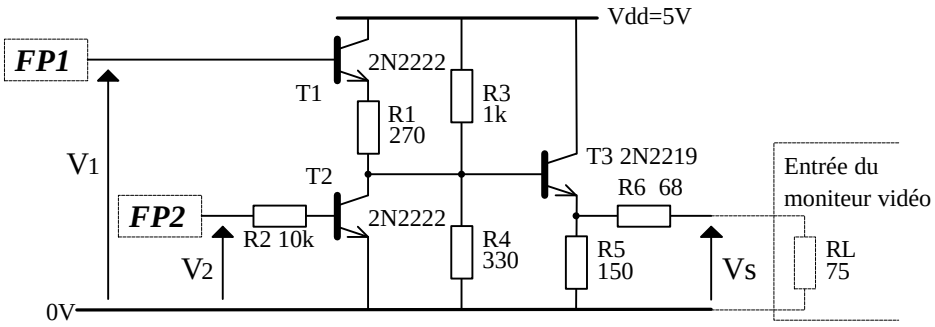
Les différentes étapes de ce travail seront expérimentées avec une maquette comportant un circuit IspLSI-1016 de LATTICE qui fournit les signaux nécessaires à un mélangeur vidéo. Celui-ci produit un signal vidéo-composite pour un moniteur monochrome MDA. Pour chacune de ces étapes, le schéma fonctionnel sera celui qui est donné ci-dessous.



Les fonctions FP1 et FP2 seront réalisées par le IspLSI-1016, programmé à partir d'un ordinateur équipé de l'outil de développement SYNARIO. La fonction FP3 est le mélangeur vidéo dont l'étude sera faite à l'étape 1 et qui sera réalisé sur une carte de câblage imprimé.

ETAPE 1 : ETUDE DE LA GENERATION DU SIGNAL VIDEO COMPOSITE (FP3)

Le schéma structurel de cette fonction est donné ci-dessous.



Le signal vidéo V_1 et le signal de synchro composite V_2 sont des tensions issues du circuit IspLSI-1016 dont les caractéristiques sont les suivantes : V_{OL} = 0V et V_{OH} = 5V

Pour les transistors bipolaires, on admet :

→ que la tension aux bornes d'une jonction passante est Vd = 0,7V ,

→ que le βmin de T1, T2, T3 vaut 250,

→ que les tensions Collecteur-Emetteur des transistors en saturation sont voisines de 0V.

1. Calculer la tension de sortie Vs pour les 4 configurations possibles des tensions V_1 et V_2, si la sortie est reliée à l'entrée d'un moniteur vidéo de résistance d'entrée R_L = 75Ω (à chaque fois, s'assurer des états de conduction de T1, T2, T3).

2. Réaliser la mise en oeuvre de cette fonction sur une carte de câblage imprimé qui aura été routée à l'aide d'un outil informatique. En vérifier la conformité avec les résultats de l'étude.

ETAPE 2 : ETUDE DE LA GENERATION DE LA SYNCHRO COMPOSITE (FP2)

Ce signal est généré par des fonctions logiques qui ont été programmées dans le IspLSI-1016.
Le fichier de description en langage ABEL de cette logique est donné ci-dessous.

```
MODULE synchro

"Inputs
CLK pin;

"Outputs
HS,S pin 22,7;
Q13..Q0 pin istype 'reg';

"Option propre au circuit ISPLSI 1016 de LATTICE
PLSI PROPERTY 'Y1_AS_RESET OFF';

"Declarations
Q = [Q13..Q0];
X = [Q4..Q0];

Equations
Q.clk = CLK;
when Q < 9983 then Q := Q+1; else Q := 0;
when X > 31-2 then HS = 1; else HS = 0;
S = HS # (Q13 & Q10 & Q9);

Test_Vectors
([CLK,!Q] -> [HS,S,Q])
[.P.,9600] -> .X.;
@repeat 400 {[.C.,.X.] -> .X.;}

END
```

3. Le moniteur vidéo utilisé possède un balayage horizontal prévu pour fonctionner avec une durée de ligne de 64µs. Après analyse et simulation de ce fichier ABEL, déterminer :
- la fréquence de l'horloge à choisir,
- la durée d'une trame, d'une image, la fréquence image,
- la durée du top de synchro ligne, du top de synchro trame,
- le nombre total de lignes composant une image, le nombre de lignes perdues dans la synchro trame.

4. Vérifier expérimentalement les caractéristiques temporelles du signal de synchro composite.

ETAPE 3 : GENERATION D'UNE IMAGE SIMPLE

5. En utilisant l'une des sorties Q13 à Q0 comme signal vidéo, on obtient une image simple. Essayer différentes possibilités et justifier l'image obtenue (la visualisation du signal vidéo composite à l'oscilloscope aide à la compréhension).

ETAPE 4 : GENERATION DE L'IMAGE D'UNE SEULE FIGURE SIMPLE (BARRE, CROIX, RECTANGLE)

Le spot balaye l'écran de manière continue à un rythme imposé par les circuits de balayage horizontaux et verticaux du moniteur.
Le signal de synchro composite permet seulement de synchroniser ces circuits de balayage avec le signal vidéo que l'on veut visualiser.
Dans le cas simple qui est traité ici, le signal vidéo ne commande que 2 états du spot (allumé ou éteint).
Pour allumer le spot en un point précis de l'écran, il faut que le signal vidéo passe à l'état haut au moment ou le spot passe par ce point et ceci à chaque balayage de l'image.
Il doit donc exister une relation entre le temps qui s'est écoulé depuis l'origine du balayage (après la synchro trame) et la position du spot sur l'écran.
Dans le cas étudié ici cette relation est contenue dans la valeur de la variable Q (c'est la valeur prise par le mot binaire formé par les sorties du compteur servant à générer les signaux de synchronisation).
Le temps nécessaire au spot pour parcourir une ligne est le temps nécessaire au comptage de 32 périodes d'horloge (Q compte de 0 à 31). Il y a donc 32 positions différentes possibles dans le cas présent (moins 2 pour le top synchro ligne) repérées chacune par la valeur de Q limitée au 5 premiers bits (on a appelé ce mot X).
Si le signal vidéo est à 1 lorsque X vaut l'une de ces positions, le spot restera allumé durant une période d'horloge. Le phénomène se reproduisant à chaque ligne, l'image formée sera celle d'une barre verticale de largeur $\frac{1}{32}$ d'écran.
Les 9 bits de poids fort de Q représentent le numéro de la ligne (0 à 137h = 311). Afin de former une barre horizontale d'épaisseur suffisament visible, on décide de faire des groupement de 8 lignes successives. Il reste alors 6 bits pour définir un paquet de 8 lignes. On appelle Y ce mot de 6 bits qui peut occuper 39 positions différentes (0 à 38).
Le tableau suivant résume ce qui vient d'être dit.

Numéro de la ligne									Numéro de la position sur une ligne				
Q13	Q12	Q11	Q10	Q9	Q8	Q7	Q6	Q5	Q4	Q3	Q2	Q1	Q0
Y (numéro d'un paquet de 8 lignes)									X				

6. Générer une image formée de la barre verticale numéro H.

7. Générer une image formée de la barre horizontale (épaisseur 8 lignes) numéro V.

8. Générer une image formée de la barre verticale numéro H et de la barre horizontale (épaisseur 8 lignes) numéro V.

9. Générer une image formée de l'intersection de la barre verticale numéro H et de la barre horizontale numéro V.

ETAPE 5 : GENERATION DE L'IMAGE D'UN RECTANGLE DEPLAÇABLE

Si l'on veut déplacer un rectangle comme celui de la question 9 de l'étape précédente, il suffit de faire varier ses coordonnées H et V. Il est alors pratique de générer ces coordonnées avec des compteurs. Dans ce cas, un déplacement continu du rectangle est obtenu par un comptage perpétuel des coordonnées H et V. Afin que ce déplacement ait lieu à une vitesse compatible avec notre vision, l'incrémentation pourra se faire une fois par balayage d'une image.

10. Générer une image formée d'un rectangle comme celui de la question 9 qui se déplace continuement horizontalement sur l'écran.

11. Générer une image formée d'un rectangle comme celui de la question 9 qui se déplace continuement verticalement sur l'écran.

12. Générer une image formée d'un rectangle comme celui de la question 9 qui se déplace continuement en diagonale sur l'écran.

ETAPE 6 : GENERATION DE L'IMAGE D'UN RECTANGLE REBONDISSANT SUR LES BORDS DE L'ECRAN

Pour réaliser l'impression d'un rebond il suffit de changer le sens du déplacement et donc du comptage. Pour ceci il faut synthétiser un compteur décompteur à butées. On donne ci-dessous la description en langage ABEL d'un circuit de ce type.

```
MODULE triangle

"Inputs
H pin;

"Outputs
Q3..Q0 pin istype 'reg'; Q = [Q3..Q0];
SENS node istype 'reg';"comptage 0; décomptage 1

"Constantes
UP = 0;   DN = 1;
BUTEE_HAUTE = 10;
BUTEE_BASSE = 5;

Equations
Q.CLK = H;
WHEN (SENS == UP) THEN Q := Q + 1; ELSE Q := Q - 1;

SENS.CLK = H;
     WHEN (Q == BUTEE_HAUTE) THEN SENS := DN;
ELSE WHEN (Q == BUTEE_BASSE) THEN SENS := UP;
                             ELSE SENS := SENS;

Test_vectors
([H,!Q] -> [SENS,Q])
@repeat 30 {[.C.,.X.] -> .X.;}
"Demarrage du compteur avec une valeur superieure aux butees
[.P.,14] -> .X.;
@repeat 30 {[.C.,.X.] -> .X.;}

END
```

13. Générer une image d'un rectangle se déplaçant en diagonale et rebondissant sur les bord de l'écran.

ELEMENTS DE REPONSE ET COMMENTAIRES

ETAPE 1

1. On donne ci-dessous un tableau récapitulatif des différents états de fonctionnement du mélangeur.

	V1	V2	T1	T2	T3	Vs
Top de synchro	0 ou 1	1	bloqué ou lin.	saturé	bloqué	0V
Spot allumé	1	0	linéaire	bloqué	linéaire	1,05V
Spot éteint	0	0	bloqué	bloqué	linéaire	0,28V

ETAPE 2

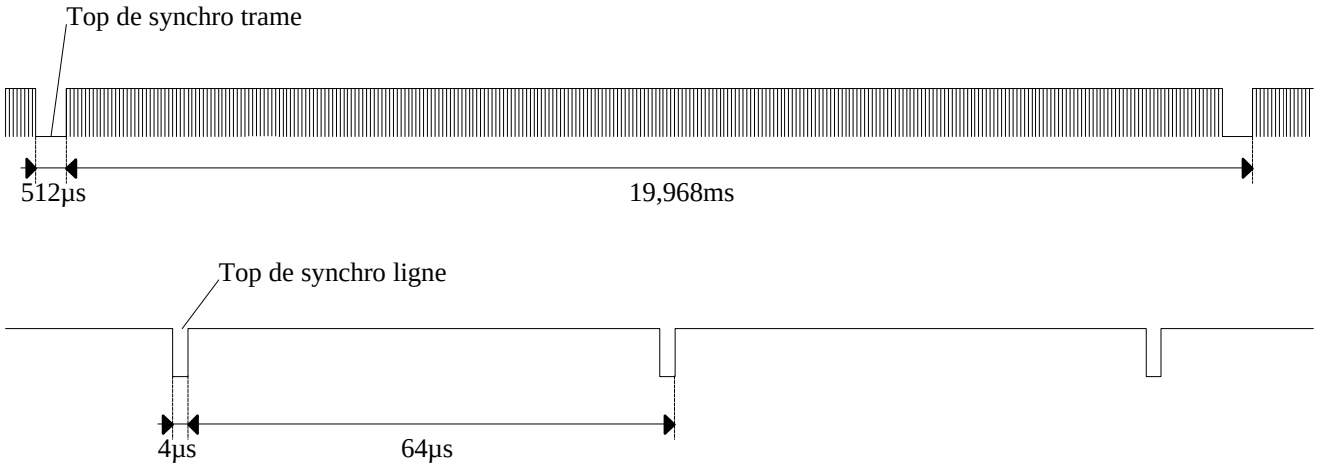
3. ➢ Dans la description en ABEL, la ligne "`when X > 31-2 then HS = 1; else HS = 0;`" décrit la génération du top de synchro ligne. `X` représente les 5 bits de poids faible du compteur `Q` d'horloge `CLK`. Il faudra $2^5 = 32$ périodes de `CLK` pour parcourir une ligne complète. La durée d'une ligne étant 64µs, la période de l'horloge devra être :
$T = \frac{64\mu s}{32} = 2\mu s$ soit une fréquence de 500kHz.

➢ Dans la description en ABEL la ligne "`when Q < 9983 then Q := Q+1; else Q := 0;`" décrit un compteur synchrone modulo 9984 d'horloge `CLK`. Le cycle de ce compteur correspond à un balayage complet de l'écran. La durée totale de l'image est égale à 9984.T soit 19,968ms. La fréquence image sera $\frac{1}{19,968ms} = 50,08Hz$ avec un balayage non entrelacé.

➢ Le signal `HS` représente la synchro horizontale qui vaut 0 tout le temps sauf lorsque `X` vaut 30 ou 31. Le top de synchro ligne dure donc 2 périodes d'horloge, soit 4µs.
La ligne "`S = HS # (Q13 & Q10 & Q9);`" décrit le signal de synchro composite. La synchro ligne est donc formée par `(Q13 & Q10 & Q9)`. La synchro trame débute lorsque Q13 = 1, Q10 = 1, Q9 = 1 les autres sorties valant 0 ce qui correspond à Q = 9728. La synchro trame se termine à la remise à 0 de Q (9984). La durée de la synchro trame sera $9984 \cdot T - 9728 \cdot T = 256 \cdot T = 8\,\text{lignes} = 512\mu s$.

➢ Une image durant 9984.T, une ligne 32.T, il y aura $\frac{9984 \cdot T}{32 \cdot T} = 312\,\text{lignes}$ dans une image dont 8 perdues dans la synchro trame.

On donne, ci-dessous, l'allure d'un cycle de balayage complet du signal de synchro-composite que l'on obtient à la sortie du mélangeur vidéo (FP3). On donne, ensuite, une vue agrandie de ce signal, limitée à la durée du balayage de 2 lignes.



ETAPE 3

5. On constate les résultats suivants :

Sortie du compteur utilisée comme signal vidéo	Image observée
Q0	15 barres verticales très fines
Q1	7 barres verticales fines
Q2	4 barres verticales
Q3	2 grosses barres verticales
Q4	1 barre verticale partie droite de l'écran
Q5	151 très fines barres horizontales
Q6	75 très fines barres horizontales
Q7	38 fines barres horizontales
Q8	19 barres horizontales
Q9	9 barres horizontales
Q10	5 barres horizontales
Q11	2 grosses barres horizontales
Q12	1 barre horizontale au milieu de l'écran
Q13	1 barre horizontale en bas de l'écran

ETAPE 4

6. Pour générer une barre verticale à la position H, il suffit que le signal vidéo passe à l'état logique 1 lorsque le numéro de la position sur une ligne vaut H donc lorsque X égale H.

7. Pour générer une barre horizontale à la position V, il suffit que le signal vidéo passe à l'état logique 1 lorsque le numéro du paquet de 8 lignes vaut V donc lorsque Y égale V.

8. Pour générer à la fois une barre verticale à la position H et une barre horizontale à la position V, il suffit que (X égale H) OU (Y égale V).

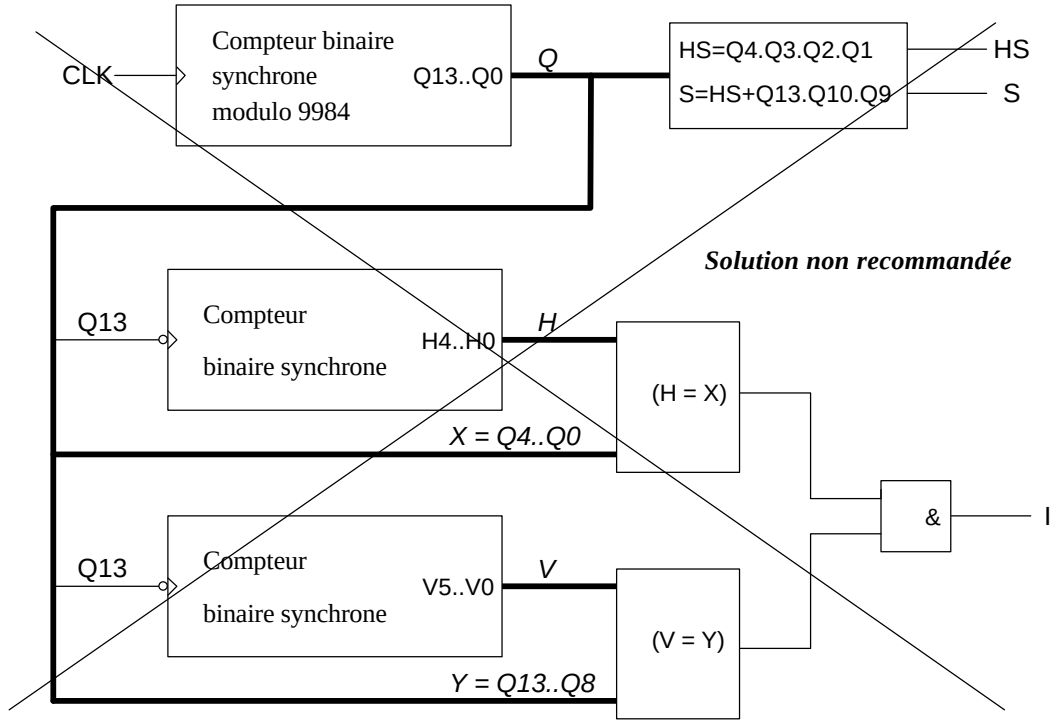
9. Pour générer l'intersection d'une barre verticale à la position H et d'une barre horizontale à la position V, il suffit que (X égale H) ET (Y égale V).

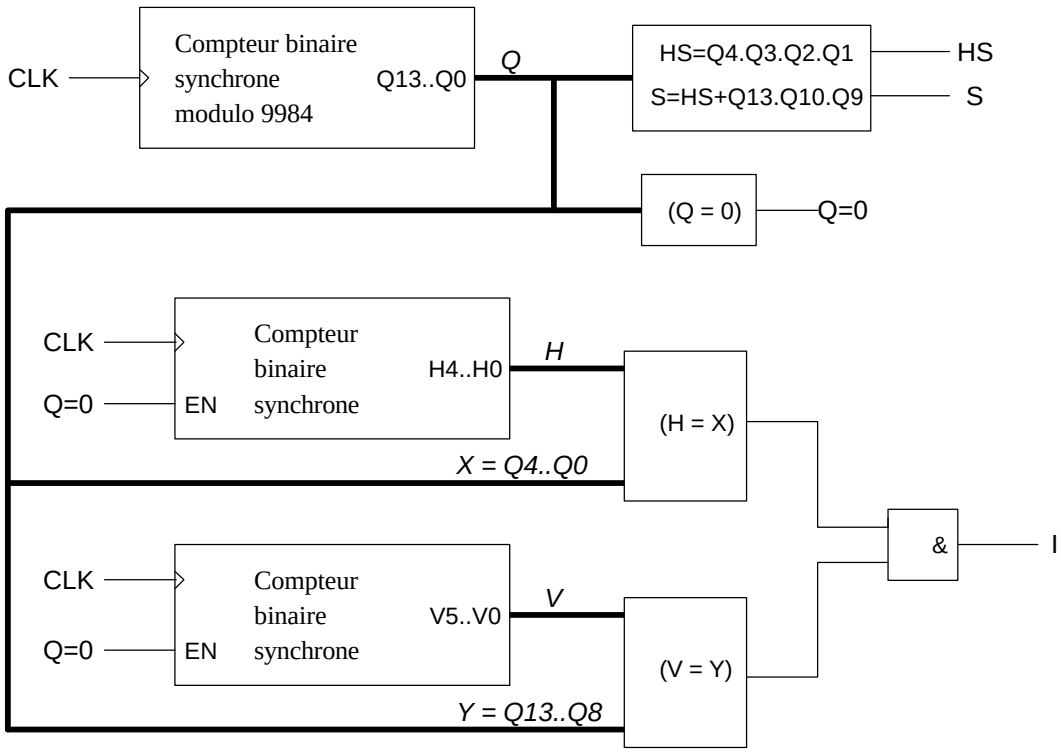
```
MODULE image1
" Generation d une barre horizontale ou verticale ou bien
" des 2 à la fois ou bien encore
" de l intersection des 2

"Inputs
CLK pin;

"Outputs
HS pin; "Synchro ligne
S,I pin 7,8; "Sorties Synchro composite et Signal vidéo
Q13..Q0 pin istype 'reg';

"Options
PLSI PROPERTY 'Y1_AS_RESET OFF';

"Declarations
Q = [Q13..Q0];
X = [Q4..Q0]; "abscisse du spot
Y = [Q13..Q8]; "ordonnée du spot

Equations
Q.clk = CLK;
when Q < 9983 then Q := Q+1; else Q := 0; "génération de la durée d'une image
when X > 31-2 then HS = 1; else HS = 0; "génération de la synchro ligne
S = HS # (Q13 & Q10 & Q9); "génération de la synchro composite

" Barre verticale
"when (X == 15) then I = 1; else I = 0;

" Barre horizontale
"when (Y == 19) then I = 1; else I = 0;

" Barre horizontale et verticale
"when (X == 15) # (Y == 19) then I = 1; else I = 0;

" Intersection des barres horizontales et verticales
when (X == 15) & (Y == 19) then I = 1; else I = 0;

Test_Vectors
([CLK , !Q] -> [HS,S,Q,I])
[.P.,9500] -> .X.; "à modifier suivant les cas
@repeat 500 {[.C.,.X.] -> .X.;}

END
```

ETAPE 5

10. 11. 12. Les compteurs définissant la position du rectangle doivent s'incrémenter à la même fréquence que le balayage complet d'une image. Le moment de cette incrémentation sera choisi durant la synchro trame.
On donne ci-dessous les synoptiques de 2 solutions pour la question 12.

- La première est celle qui sera probablement proposée par les élèves. Dans cette solution, les compteurs définissant la position du rectangle ont une liaison asynchrone avec les circuits de génération des synchronisations. D'une manière générale cette façon de procéder peut amener à des fonctionnements erratiques difficilement décelables, il est préférable de prendre l'habitude de ne pas l'utiliser.
- Dans la seconde tous les compteurs ont un fonctionnement synchrone avec l'horloge CLK. Cette solution est préférable d'autant que dans un circuit logique programmable elle occupe la même quantité de ressources.



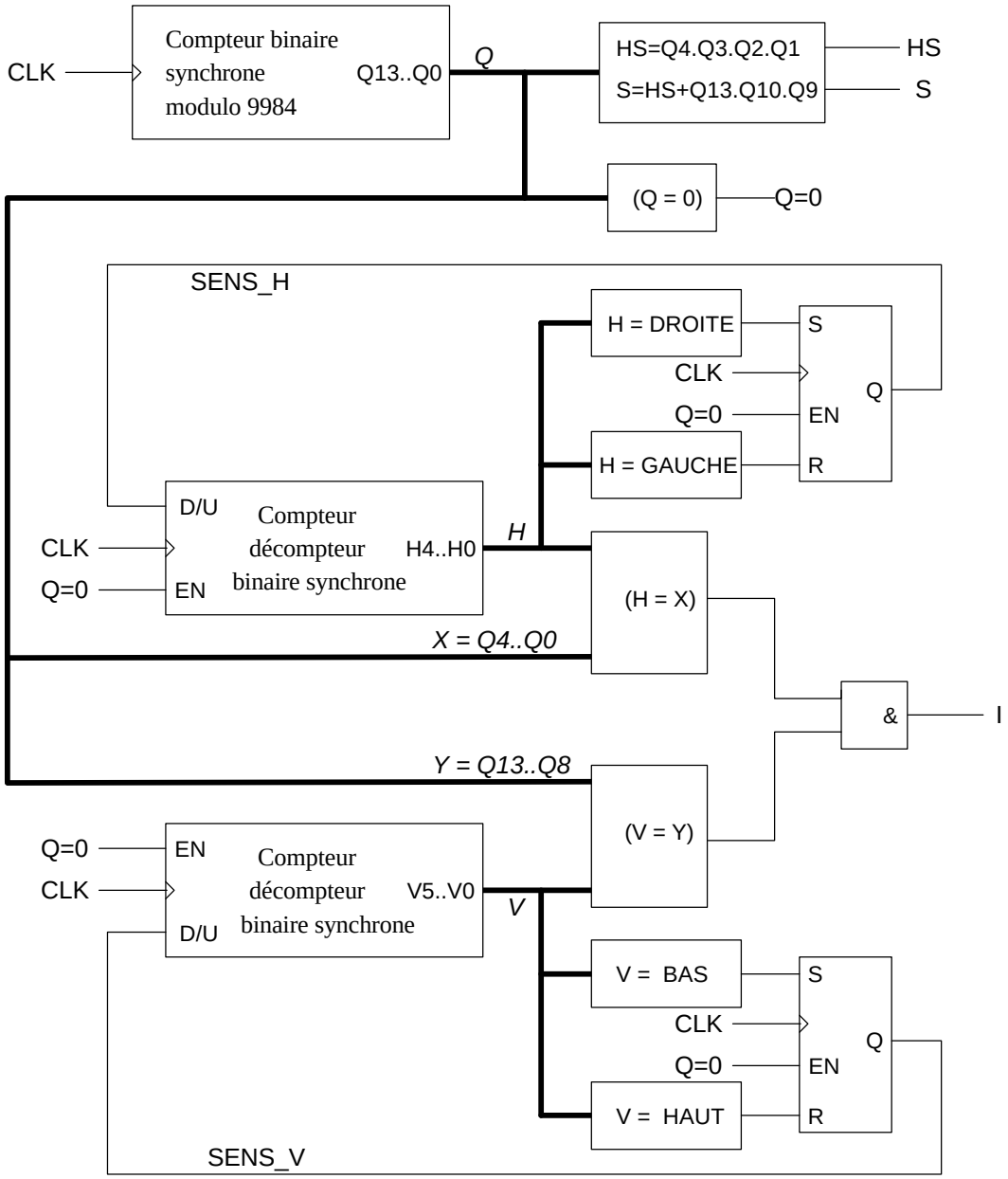


On donne ci-dessous la traduction du deuxième synoptique en langage ABEL.

```
MODULE image2
"Generation d un rectangle se deplacant en oblique vers le bas a droite

"Inputs
CLK pin;

"Outputs
HS node; "Synchro ligne
S,I pin 7,8; "Sorties Synchro composite et Signal vidéo
Q13..Q0 node istype 'reg';
H4..H0,V5..V0 node istype 'reg';

"Options
PLSI PROPERTY 'Y1_AS_RESET OFF';

"Declarations
Q = [Q13..Q0];
X = [Q4..Q0]; "abscisse du spot
Y = [Q13..Q8]; "ordonnée du spot
H = [H4..H0]; "position horizontale du rectangle
V = [V5..V0]; "position verticale du rectangle

Equations

"Instruction limitant la complexite des equations et facilitant le routage
@carry 4;

"Synchronisation
Q.clk = CLK;
when Q < 9983 then Q := Q+1; else Q := 0; "génération de la durée d'une image
when X > 31-2 then HS = 1; else HS = 0; "génération de la synchro ligne
S = HS # (Q13 & Q10 & Q9); "génération de la synchro composite

"Définition de la position horizontale du rectangle
H.clk = CLK;
when Q==0 then H:=H+1; else H:=H;

"Définition de la position verticale du rectangle
V.clk = CLK;
when Q==0 then V:=V+1; else V:=V;

"Génération de l'image
when (X == H) & (Y == V) then I = 1; else I = 0; "génération du rectangle

END
```

ETAPE 6

13. De la même manière qu'à l'étape précédente, les élèves auront tendance à proposer une solution asynchrone. Il faudra les inciter à travailler sur une solution totalement synchrone comme celle donnée ci-dessous.
La description en langage ABEL de cette solution est donné à la page suivante.

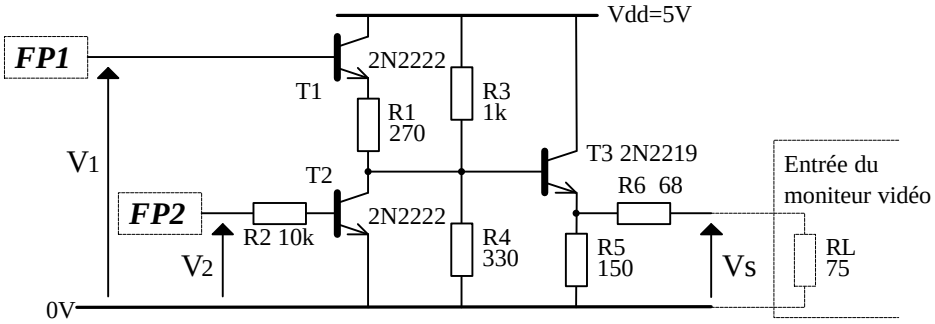


```
MODULE image3
"Generation d un rectangle se deplacant en oblique et
"rebondissant sur les bords de l ecran

"Inputs
CLK pin;

"Outputs
HS node; "Synchro ligne
S,I pin 7,8; "Sorties Synchro composite et Signal vidéo
Q13..Q0 node istype 'reg';
H4..H0,V5..V0 node istype 'reg';
SENS_H,SENS_V node istype 'reg';"sens de déplacement hor. et vert. du rect.

"Options
PLSI PROPERTY 'Y1_AS_RESET OFF';

"Declarations
Q = [Q13..Q0];
X = [Q4..Q0]; "abscisse du spot
Y = [Q13..Q8]; "ordonnée du spot
H = [H4..H0]; "position horizontale du rectangle
V = [V5..V0]; "position verticale du rectangle
DROITE = 29; GAUCHE = 2;"bords horizontaux
BAS = 37; HAUT = 0;"bords verticaux
ADROITE = 0; AGAUCHE = 1;
ENBAS = 0; ENHAUT = 1;

Equations

"Instruction limitant la complexite des equations et facilitant le routage
@carry 4;

"Synchronisation
Q.clk = CLK;
when Q < 9983 then Q := Q+1; else Q := 0; "génération de la durée d'une image
when X > 31-2 then HS = 1; else HS = 0; "génération de la synchro ligne
S = HS # (Q13 & Q10 & Q9); "génération de la synchro composite

"definition de la position horizontale du rectangle
H.clk = CLK;
     when (Q != 0)            then H := H;
else when (SENS_H == ADROITE) then H := H+1;
                              else H := H-1;

SENS_H.clk = CLK;
     when (H == DROITE) then SENS_H := AGAUCHE;
else when (H == GAUCHE) then SENS_H := ADROITE;
                        else SENS_H := SENS_H;

"definition de la position verticale du rectangle
V.clk = CLK;
     when (Q != 0)          then V := V;
else when (SENS_V == ENBAS) then V := V+1;
                            else V := V-1;

SENS_V.clk = CLK;
     when (V == BAS)  then SENS_V := ENHAUT;
else when (V == HAUT) then SENS_V := ENBAS;
                      else SENS_V := SENS_V;

"Génération de l'image
when (X == H) & (Y == V) then I = 1; else I = 0; "génération du rectangle

END
```

EVALUATION

GENERATION DU SIGNAL VIDEO COMPOSITE (FP3)

Le schéma structurel de cette fonction est donné ci-dessous.



La sortie du montage est reliée à l'entrée d'un moniteur vidéo de résistance d'entrée $R_L = 75\Omega$
Le signal vidéo V_1 et le signal de synchro composite V_2 sont des tensions issues du circuit IspLSI-1016 dont les caractéristiques sont les suivantes : $V_{OH} \geq 2,4V$ et $V_{OL} \leq 0,4V$.
Pour les transistors bipolaires, on admet :

→ que la tension aux bornes d'une jonction passante est Vd = 0,7V ,
→ que le βmin de T1, T2, T3 vaut 250,
→ que les tensions Collecteur-Emetteur des transistors en saturation sont voisines de 0V.

1. Calculer la tension de sortie Vs pour les 4 configurations possibles des tensions V_1 et V_2, si :
* V_{OH} =5V et V_{OL} = 0V puis
* V_{OH} = 2,4V et V_{OL} = 0,4V

(à chaque fois, s'assurer des états de conduction de T1, T2, T3).

2. Quels niveaux logiques faudra-t-il placer sur V_1 et V_2 pour obtenir :
- un top de synchronisation ligne ou trame,
- un spot "allumé" (niveau de blanc),
- un spot "éteint" (niveau de noir).

A quelles valeurs de la tension Vs correspondent ces différents cas ?
Le signal vidéo-composite fourni sera-t-il exploitable par le moniteur quelles que soient les valeurs de V_{OH} et V_{OL} (dans les limites fournies ci-dessus) ?

GENERATION DE LA SYNCHRO COMPOSITE (FP2)

Ce signal est généré par des fonctions logiques qui ont été programmées dans le IspLSI-1016.
Le fichier de description en langage ABEL de cette logique est donné ci-dessous pour un moniteur vidéo possèdant un balayage horizontal prévu pour fonctionner avec une durée de ligne de 64µs.

```
MODULE synchro
CLK pin;
HS,S pin 22,7;
Q13..Q0 pin istype 'reg';
Q = [Q13..Q0];
X = [Q4..Q0];

Equations
Q.clk = CLK;
when Q < 9983 then Q := Q+1; else Q := 0;
when X > 31-2 then HS = 1; else HS = 0;
S = HS # (Q13 & Q10 & Q9);
END
```

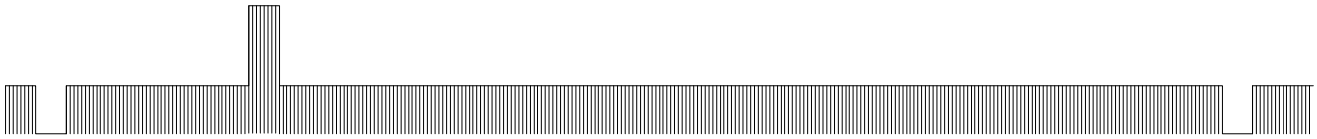
3. Déterminer en donnant les justifications :
- la fréquence de l'horloge à choisir,
- la durée d'une trame, d'une image, la fréquence image,
- la durée du top de synchro ligne, du top de synchro trame,
- le nombre total de lignes composant une image, le nombre de lignes perdues dans la synchro trame.

4. Modifier le fichier de description en langage ABEL permettant de générer le signal de synchro composite si la fréquence de l'horloge du circuit logique IspLSI-1016 est : | 250kHz | | 1MHz | | 2MHz |

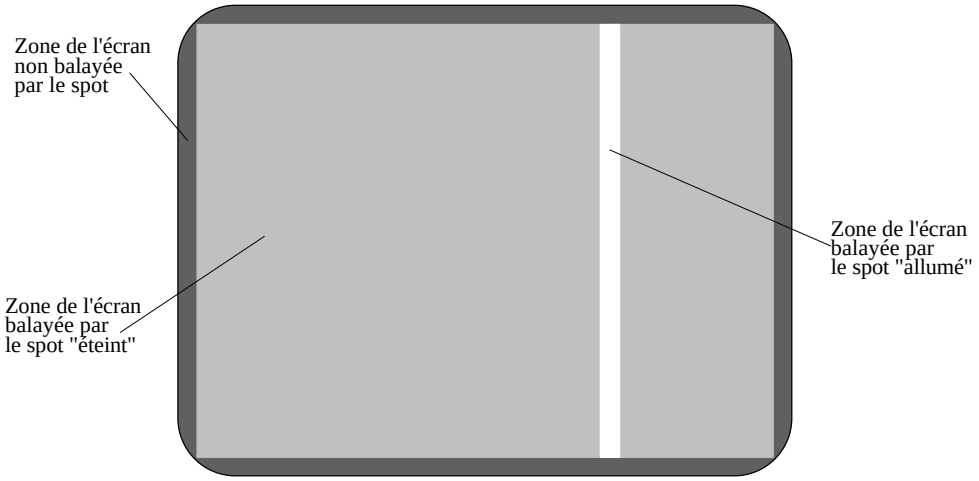
GENERATION D'UNE IMAGE SIMPLE

Pour cette partie, on admet que le moniteur permet de visualiser l'intégralité du signal vidéo hormis les durées de synchronisation.

5. Dessiner l'image générée par le signal vidéo-composite donné ci-dessous. En préciser la forme et les dimensions relatives.



6. Donner l'oscillogramme d'une trame du signal vidéo-composite qui forme l'image donnée ci-dessous. Fournir une vue détaillée de l'oscillogramme pour une ligne. Déterminer approximativement les différentes durées.



7. Dessiner l'image générée par la description en langage ABEL donnée ci-dessous. Préciser la forme, les dimensions et la position de la figure générée.

```
MODULE image

CLK pin;
HS pin;
S,I pin 7,8; "Sorties Synchro composite et Signal vidéo
Q13..Q0 pin istype 'reg';

Q = [Q13..Q0];
X = [Q4..Q0]; "abscisse du spot
Y = [Q13..Q8]; "ordonnée du spot

Equations
Q.clk = CLK;
when Q < 9983 then Q := Q+1; else Q := 0;
when X > 31-2 then HS = 1; else HS = 0;
S = HS # (Q13 & Q10 & Q9);
when (X == 15) & (Y == 19) then I = 1; else I = 0;
END
```

8. Dessiner le schéma de câblage des différents sous-ensembles utilisés ici pour visualiser l'image de la question précédente sur l'écran du moniteur.
Quel est le rôle du fichier "ABEL" ? Pourquoi son contenu porte-t-il le nom de "**description**" ?
Quelles sont les fonctions remplies par les outils logiciels utilisés ?
Justifier, qu'une fois l'image obtenue sur l'écran, on puisse déconnecter l'ordinateur.

CHOIX MATERIELS

Le travail proposé dans ce document a été effectué par les élèves de BTS électronique 2° année du lycée Gustave Eiffel de GAGNY avec les conditions matérielles suivantes :

- 6 ordinateurs équipés du logiciel de développement SYNARIO (version gratuite limitée à la série IspLSI-1000 et 2000 de LATTICE). SYNARIO permet de mélanger une description sous forme schématique et des descriptions en langage ABEL et il dispose d'un simulateur fonctionnel.
- 6 maquettes universelles comportant, outre une petite alimentation 5V, 1 circuit IspLSI-1016 de LATTICE, une horloge à quartz et un diviseur de fréquence (74HC4060).
- 6 câbles de téléchargement reliant le port parallèle d'un ordinateur à la maquette ci-dessus.
- 6 moniteurs CGA-MDA.

CHOIX DU CIRCUIT LOGIQUE PROGRAMMABLE

Inventaire des ressources nécessaires

- La génération des signaux de synchronisation utilise un compteur synchrone à 14 étages qui nécessite au minimum 14 registres (bascules D ou T).
- Les compteurs synchrones générant la position du rectangle nécessitent respectivement au minimum 5 et 6 registres.
- Les bascules RS gérant le sens de déplacement du rectangle utilisent 2 registres à elles deux.
- Reste environ 6 noeuds de type combinatoire qui vont utiliser au moins 6 sorties de cellule (6 registres sont donc bloqués en mode combinatoire).

On arrive au minimum à un circuit contenant 33 registres.
Effectivement, l'évaluation avant le routage donne l'utilisation de 9 GLB soit 36 registres et le bilan post routage donne l'utilisation de 14 GLB soit l'équivalent de 56 registres. Le circuit est plein à 85%.
Si l'on envisage une augmentation de la finesse des dessins sur l'écran, on devra augmenter le nombre de bits nécessaires pour définir la position des objets et donc augmenter la dimension des compteurs. Il faut alors rapidement envisager d'utiliser un circuit présentant plus de ressources. Par exemple, en utilisant une horloge à 1MHz au lieu de 500kHz, on atteint la limite des possibilités de l'IspLSI-1016 lors de l'étape 6.

Circuits ayant fait l'objet d'une expérimentation

En dehors du circuit IspLSI-1016 de LATTICE (CPLD de 64 registres), ont été expérimenté :

- Le circuit CY7C374i de CYPRESS programmé avec le logiciel *WARP2* ISR (version d'évaluation gratuite). Le langage de description utilisé est le VHDL sans utilisation de schématique. Le logiciel dispose d'un simulateur fonctionnel post routage. Le circuit CY7C374i est aussi un circuit programmable directement sur la maquette.
  Sa structure interne de type CPLD permet l'utilisation de 128 registres (bascules D ou T).
  On fournit, en annexe, les descriptions en langage VHDL pour ce circuit, demandées aux étapes 4, 5 et 6.

- Le circuit PZ 5128 de PHILIPS programmé *in situ* à travers une liaison au standard JTAG par le logiciel XPLA (version d'évaluation gratuite). Le langage de description utilisé, le PHDL, est très proche du langage ABEL. Le logiciel dispose d'un simulateur fonctionnel post routage. Il n'est pas non plus prévu d'utilisation de schématique.
  La structure interne de ce circuit, de type CPLD, permet l'utilisation de 128 registres (bascules D ou T) avec une utilisation optimum des ressources combinatoires grâce à une organisation des cellules de type FPLA (matrice ET et matrice OU programmables).
  En outre, cette famille de circuits présente la particularité, rare pour des CPLD, d'avoir une consommation statique très faible (100µA).

- Le circuit XILINX utilisé lors d'un récent projet, le XC3020A qui dispose de 64 Configurable Logic Blocs (CLB) comprenant chacun 2 registres soit 128 registres aurait pu convenir. Or il possède, de par sa structure de type FPGA, de faibles ressources combinatoires dans chaque CLB. Il doit donc utiliser de nombreux CLB (et donc des registres) pour réaliser les parties combinatoires. **Ce circuit n'est absolument pas utilisable ici**, on n'arrive même pas à y router la génération des synchros.
  Sur les cartes "Demo Board" de XILINX, on dispose aussi d'un XC4003 (100 CLB à 2 registres) qui **ne convient pas non plus**. Le travail proposé ici, ne peut pas y être routé : à moins, peut être, d'être un expert en la matière.
  Par contre, si l'on souhaite utiliser le couple XACT-VIEWLOGIC (ou FOUNDATION), on pourra certainement (non vérifié actuellement) se tourner vers les CPLD de XILINX tels que le XC7354 et le XC7272.

La liste des circuits possibles n'est bien entendu pas limitative :

- ALTERA, le premier fabricant mondial de circuits logiques programmables, en terme de quantité de circuits vendus, propose de nombreuses références utilisables. Malheureusement, le logiciel de développement (MAX+2) semble assez onéreux.
- VANTIS anciennement AMD, propose des CPLD programmables *in situ* (série MACH) avec des logiciels de développement gratuits (soit une version limitée de SYNARIO, soit un logiciel de VANTIS : DESIGN-DIRECT).

Le lycée de GAGNY a fait réaliser des maquettes pour les circuits LATTICE. Nous pouvons vous fournir les documents de fabrication (typons) de ces maquettes. Vous pouvez nous contacter à :

Mr BONNET
Lycée Gustave Eiffel
16 chemin de la Renardière
93220 GAGNY
☎: 01 43 02 80 36
Fax : 01 43 02 82 14

CHOIX DU MONITEUR

Le choix du moniteur va dépendre essentiellement du matériel disponible.

- Dans les travaux présentés ici, on a utilisé d'anciens moniteurs CGA monochromes ou MDA munis d'une liaison par fil blindé et fiche CINCH (durée de ligne 64µs, balayage vertical à 50Hz).
  Cette liaison transporte un signal vidéo composite qui est compatible avec la prise péritel des téléviseurs (bornes VIDEO IN [20] et VIDEO GROUND [18] de la prise péritel).

- Si l'on dispose de moniteurs HERCULE, le mélangeur de FP3 n'est plus nécessaire. Les caractéristiques temporelles des signaux de synchronisations sont les mêmes que pour les moniteur MDA. Il suffit de sortir directement du circuit logique (niveau TTL) :
  - le signal de synchronisation trame (synchro active à l'état bas) (broche 9 de la DB9),
  - le signal de synchronisation ligne (synchro active à l'état bas) (broche 8 de la DB9),
  - le signal vidéo (vidéo positive) (broche 7 de la DB9 et la masse broche 1).

  Attention : en l'absence d'un signal de synchro ligne correct le moniteur HERCULE ne balaie pas l'écran.

- Si l'on dispose de moniteur VGA, il est tout à fait possible d'envisager des applications en couleurs. Dans ce cas, le circuit logique attaquera le moniteur à travers des résistances de 68Ω pour les synchros et quelques centaines d'ohms pour les signaux vidéo. Pour la couleur, en plus des 2 signaux de synchronisation (actifs à l'état bas), il faudra fournir les signaux vidéo des 3 couleurs de base (Rouge, Vert, Bleu).
  Les caractéristiques temporelles des signaux de synchronisation devront être adaptées au moniteur utilisé.
  Pour le mode VGA de base (640x480) la fréquence ligne est d'environ 31,5kHz pour une fréquence trame de 60Hz.
  On donne en annexe, la description en langage ABEL relative à l'étape 6 pour un moniteur VGA.
  Le branchement à la prise DB15 du moniteur VGA sera le suivant :
  - Signal vidéo Rouge (0,7V sur 75Ω) : broche 1,
  - Signal vidéo Vert (0,7V sur 75Ω) : broche 2,
  - Signal vidéo Bleu (0,7V sur 75Ω) : broche 3,
  - Signal de synchro ligne (TTL sur 75Ω) : broche 13,
  - Signal de synchro trame (TTL sur 75Ω) : broche 14,
  - Masse : broche 6, 7, 8, 10.

ANNEXES

DESCRIPTIONS EN LANGAGE VHDL (MONITEUR MDA-CGA)

Etape 4 : génération de l'image d'une barre horizontale ou verticale, d'une croix ou d'un rectangle

```
library ieee;

use ieee.std_logic_1164.all;
use work.std_arith.all;

entity IMAGE1 is
port (CLK : in std_logic;
      S,I : out std_logic);
attribute pin_numbers of IMAGE1:entity is
"CLK:20 S:3 I:4";
end IMAGE1;

architecture ARCH_IMAGE1 of IMAGE1 is
signal Q : std_logic_vector(13 downto 0);
signal X : std_logic_vector(4 downto 0);
signal Y : std_logic_vector(5 downto 0);
signal HS : std_logic;
begin
        X <= Q(4 downto 0);
        Y <= Q(13 downto 8);
        process (CLK)
                begin
                if (CLK'event and CLK='1') then
                        if Q < x"26FF" then Q <= Q+1;
                        else Q <= "00000000000000";
                        end if;
                end if;
        end process;
        HS <= '1' when X > x"1F" - x"2" else '0';
        S <= HS or (Q(13) and Q(10) and Q(9));

--barre horizontale
--      I <= '1' when X = "01111" else '0';

--barre verticale
--      I <= '1' xhen Y = "010011" else '0';

--barre hor. et vert.
        I <= '1' when (X = "01111") or (Y = "010011") else '0';

--rectangle
--      I <= '1' when (X = "01111") and (Y = "010011") else '0';

end ARCH_IMAGE1;
```

Etape 5 : génération de l'image d'un rectangle déplaçable

```
library ieee;
use ieee.std_logic_1164.all;
use work.std_arith.all;

entity IMAGE2 is
port (CLK : in std_logic;
      S,I : out std_logic);
attribute pin_numbers of IMAGE2:entity is
"CLK:20 S:3 I:4";
end IMAGE2;

architecture ARCH_IMAGE2 of IMAGE2 is
signal Q : std_logic_vector(13 downto 0);
signal X : std_logic_vector(4 downto 0);
signal Y : std_logic_vector(5 downto 0);
signal H : std_logic_vector(4 downto 0);
signal V : std_logic_vector(5 downto 0);
signal HS : std_logic;
begin
        X <= Q(4 downto 0);
        Y <= Q(13 downto 8);
        process (CLK)
                begin
                if (CLK'event and CLK='1') then

--compteur de generation des signaux de synchro
                        if Q < x"26FF" then Q <= Q+1;
                        else Q <= "00000000000000";
                        end if;

--definition de la position horizontale du rectangle
                        if Q=0 then H <= H+1; else H <= H; end if;

--definition de la position verticale du rectangle
                        if Q=0 then V <= V+1; else V <= V; end if;

                end if;
        end process;

--signal de synchro composite
        HS <= '1' when X > x"1F" - x"2" else '0';
        S <= HS or (Q(13) and Q(10) and Q(9));

--generation de l image du rectangle
        I <= '1' when (X = H) and (Y = V) else '0';

end ARCH_IMAGE2;
```

Etape 6 : génération de l'image d'un rectangle rebondissant sur les bords de l'écran

```
library ieee;
use ieee.std_logic_1164.all;
use work.std_arith.all;

entity IMAGE3 is
port (CLK : in std_logic;
      S,I : out std_logic);
attribute pin_numbers of IMAGE3:entity is
"CLK:20 S:3 I:4";
end IMAGE3;

architecture ARCH_IMAGE3 of IMAGE3 is
signal Q : std_logic_vector(13 downto 0);
signal X : std_logic_vector(4 downto 0);
signal Y : std_logic_vector(5 downto 0);
signal H : std_logic_vector(4 downto 0);
signal V : std_logic_vector(5 downto 0);
signal SENS_H : std_logic;
signal SENS_V : std_logic;
signal HS : std_logic;
constant DROITE : std_logic_vector(4 downto 0):= "11100";
constant GAUCHE : std_logic_vector(4 downto 0):= "00011";
constant BAS : std_logic_vector(5 downto 0):= "100100";
constant HAUT : std_logic_vector(5 downto 0):= "000001";
constant ADROITE : std_logic := '0';
constant AGAUCHE : std_logic := '1';
constant ENBAS : std_logic := '0';
constant ENHAUT : std_logic := '1';
begin
        X <= Q(4 downto 0);
        Y <= Q(13 downto 8);
        process (CLK)
                begin
                if (CLK'event and CLK='1') then

--compteur de generation des signaux de synchro
                        if Q < x"26FF" then Q <= Q+1;
                        else Q <= "00000000000000";
                        end if;

--definition de la position horizontale du rectangle
        --compteur decompteur
                                          if Q/=0 then H <= H;
                        else if SENS_H = ADROITE then H <= H+1;
                                                 else H <= H-1; end if;
                        end if;

        --memoire du sens de comptage
                        if Q=0 then
                                  if H = DROITE then SENS_H <= AGAUCHE;
                                elsif H = GAUCHE then SENS_H <= ADROITE;
                                                 else SENS_H <= SENS_H; end if;
                        end if;

--definition de la position verticale du rectangle
        --compteur decompteur
                                        if Q/=0 then V <= V;
                        else if SENS_V = ENBAS then V <= V+1;
                                               else V <= V-1; end if;
                        end if;
```

```
        --memoire du sens de comptage
                        if Q=0 then
                                  if V = BAS  then SENS_V <= ENHAUT;
                                elsif V = HAUT then SENS_V <= ENBAS;
                                               else SENS_V <= SENS_V; end if;
                        end if;

                end if;
        end process;

--signal de synchro composite
        HS <= '1' when X > x"1F" - x"2" else '0';
        S <= HS or (Q(13) and Q(10) and Q(9));

--generation de l image du rectangle
        I <= '1' when (X = H) and (Y = V) else '0';

end ARCH_IMAGE3;
```

DESCRIPTION EN LANGAGE ABEL (MONITEUR VGA) POUR L'ETAPE 6

Pour s'approcher des caractéristiques temporelles des signaux de synchronisation du mode VGA, on utilise une horloge à 1MHz, un compteur synchrone modulo 16640 (soit 15 bits). La fréquence trame est donc de $\frac{1}{16640\mu s} = 60,096Hz$ très proche de 60Hz. La durée d'une ligne sera de 32µs soit une fréquence ligne de $\frac{1}{32\mu s} = 31,25kHz$. Le top de synchro ligne dure 2µs et le top de synchro trame 100µs.
D'autres combinaisons sont, évidemment, possibles en particuliers avec des moniteurs VGA multi-synchro.

On donne à la page suivante la description en langage ABEL correspondant au travail de l'étape 6, adapté à l'utilisation d'un moniteur VGA.

```
MODULE VGA3
"Generation d un rectangle se deplacant en oblique et
"rebondissant sur les bords d un ecran VGA

"Inputs
CLK pin 11;

"Outputs
HS pin 17; "Synchro ligne
VS, I pin 15,16; "Sorties Synchro composite et Signal vidéo
Q14..Q0 node istype 'reg';
H4..H0,V5..V0 node istype 'reg';
SENS_H,SENS_V node istype 'reg';"sens de déplacement hor. et vert. du rect.

"Options
PLSI PROPERTY 'Y1_AS_RESET OFF';

"Declarations
Q = [Q14..Q0];
X = [Q4..Q0]; "abscisse du spot
Y = [Q14..Q9]; "ordonnée du spot
H = [H4..H0]; "position horizontale du rectangle
V = [V5..V0]; "position verticale du rectangle
DROITE = 29; GAUCHE = 2;"bords horizontaux
BAS = 31; HAUT = 1;"bords verticaux
ADROITE = 0; AGAUCHE = 1;
ENBAS = 0; ENHAUT = 1;

Equations
@carry 4;

"Synchronisation (60Hz-32us => 520 lignes)
Q.clk = CLK;"Horloge 1MHz
when Q < 16639 then Q := Q+1; else Q := 0; "génération de la durée d'une image
when X > 31-2 then HS = 0; else HS = 1; "génération de la synchro ligne
VS = !Q14;"génération de la synchro trame

"definition de la position horizontale du rectangle
H.clk = CLK;
     when (Q != 0)            then H := H;
else when (SENS_H == ADROITE) then H := H+1;
                              else H := H-1;

SENS_H.clk = CLK;
     when (H == DROITE) then SENS_H := AGAUCHE;
else when (H == GAUCHE) then SENS_H := ADROITE;
                        else SENS_H := SENS_H;

"definition de la position verticale du rectangle
V.clk = CLK;
     when (Q != 0)          then V := V;
else when (SENS_V == ENBAS) then V := V+1;
                            else V := V-1;

SENS_V.clk = CLK;
     when (V == BAS)  then SENS_V := ENHAUT;
else when (V == HAUT) then SENS_V := ENBAS;
                      else SENS_V := SENS_V;

"Génération de l'image
when (X == H) & (Y == V) then I = 1; else I = 0; "génération du rectangle

END
```

LES DIFFERENTES CATEGORIES DE CIRCUITS LOGIQUES PROGRAMMABLES

Circuits logiques programmables sur site

Réseaux logiques combinatoires programmables

Ces réseaux sont constitués de 2 matrices : une matrice ET recevant les entrées et leurs compléments et dont les sorties attaquent une matrice OU qui fournit les sorties.
Lorsqu'elles sont programmables ces matrices utilisent des fusibles (rétrécissement d'une métallisation) en série avec les composants de connexion (diodes, transistors bipolaires ou transistors à effet de champ "MOS").

Dans cette catégorie, on distingue suivant leur organisation interne :

- les mémoires PROM - EPROM - EEPROM (*Electricaly Erasable Programmable Read Only Memory*) : la matrice ET est figée (c'est le décodeur d'adresse) et c'est la matrice OU qui est programmable.
- les circuits PAL (*Programmable Array Logic*) en technologie TTL Schottky ou PLD (*Programmable Logic Devices*) en technologie CMOS : c'est l'inverse des mémoires, la matrice ET est programmable et la matrice OU est figée.
- les circuits PLA ou FPLA (*Field Programmable Logic Arrays*) : dans ces circuits les matrices ET et OU sont toutes les 2 programmables.

On donne page 27 des exemples de réseaux logiques combinatoires programmables à 2 entrées et 2 sorties utilisant des matrices à diodes.

Réseaux logiques séquentiels programmables

Ces réseaux sont des évolutions des PAL et PLD auxquelles on a ajouté en sortie des cellules, programmables ou non, de type registre (bascule D) dont les sorties sont réintroduites avec les entrées dans la matrice ET.
Dans cette catégorie, on distingue suivant la technologie utilisée :

- les PAL ou PLD séquentiels : il s'agit de circuits dont les matrices programmables sont de type à fusibles (diode ou transistor bipolaire ou MOS laissés intacts ou détruits) donc non reprogrammables.
- les EPLD ou EEPLD ou GAL (*Generic Array Logic*) : il s'agit de circuits proches des PLD séquentiels mais dont les matrices programmables sont de type EPROM (transistor MOS à enrichissement dont la grille est flottante, effaçable grâce à un rayonnement ultraviolet) ou de type EEPROM (transistor MOS à enrichissement dont la grille est flottante, effaçable grâce à une impulsion électrique).

Matrices programmables

Ces réseaux sont constitués de plusieurs cellules logiques programmables identiques disposées en matrice et interconnectables entre elles. Chaque cellule est plus ou moins inspirée de la structure des cellules de PLD séquentiels. La programmation de ces circuits consiste à programmer chacune des cellules de la matrice puis à les interconnecter. Chaque fabricant utilise des structures qui lui sont propres et il fournit donc les outils de programmation associés.

On distingue 2 types de circuits à matrices programmables suivant la conception des cellules logiques :

- les CPLD (*Complex PLD*) qui utilisent des cellules logiques d'environ 16 entrées dont les possibilités combinatoires sont fournies par une matrice ET (utilisation de *Product Terms*).

  Les principaux fabricants de ces circuits sont :

  - XILINX : Ce fabricant réalise des circuits de type EPLD (effaçable aux UV) équivalents à environ 4 à 16 GAL standards (série XC7200 et XC7300). Leur programmation doit être faite sur un programmateur spécial donc hors de la maquette (de même que leur effacement). XILINX réalise maintenant des circuits programmables directement sur la maquette.
  - ALTERA : Ce fabricant réalise des circuits de type EPLD et EEPLD (effaçable électriquement) (série Classic, MAX5000, MAX7000). Leur programmation doit être faite sur un programmateur spécial donc hors de la maquette (de même que leur effacement). Depuis peu ALTERA réalise des circuits programmables directement sur la maquette (série MAX9000, FLASHlogic)
  - LATTICE : Ce fabricant (créateur des GAL) réalise des circuits de type EEPLD, mais programmables directement sur la maquette grâce à quelques bornes du circuit, dédiées à une liaison série (série ispLSI1000, ispLSI2000, isp3000 et ispLSI6000).
  - A.M.D. : Ce fabricant réalise des circuits de type EPROM et EEPROM consistant en l'association de GAL autour d'une matrice d'interconnexion (série MACH120, MACH130).

- les FPGA (*Field Programmable Gates Array*) qui utilisent des cellules logiques à 4 ou 5 entrées dont les possibilités combinatoires sont fournies par la lecture d'une table de vérité (mémoire de 2^4 mots de 1 ou 2 bits : *Look-Up Table*). Cette approche permet une plus grande densité de cellules logiques et est utilisée pour les applications nécessitant un grand nombre de registres.

  Les principaux fabricants de ces circuits sont :

  - XILINX : Ce fabricant réalise les parties programmables des cellules logiques et les réseaux d'interconnexion avec une technologie de type SRAM (*Static Random Accesss Memory*) donc volatiles (série XC2000, XC3000 et XC4000). Ces circuits doivent être reprogrammés après chaque mise sous tension (ceci est généralement fait à partir d'une EPROM ou EEPROM implantée conjointement au FPGA). Cette technologie permet d'atteindre une grande densité d'intégration et une reprogrammation aisée, y compris directement sur la maquette. L'inconvénient est la nécessité d'un composant supplémentaire (EPROM et sa circuiterie de chargement du FPGA).
  - ALTERA : Ce fabricant réalise des circuits du même type que ceux de XILINX (SRAM) (série FLEX8000 et FLEX10K).
  - ACTEL : Ce fabricant réalise la partie programmable de ses cellules logiques ainsi que les réseaux d'interconnexion avec la technologie dite d'"anti-fusible" ; il s'agit d'une zone isolante très fine séparant 2 conducteurs qui, au départ est non conductrice et que la programmation détruit en la claquant ce qui la rend conductrice (à l'inverse d'un fusible). Ces circuits permettent d'obtenir de grandes densités d'intégration, ils sont non volatiles et non reprogrammables.

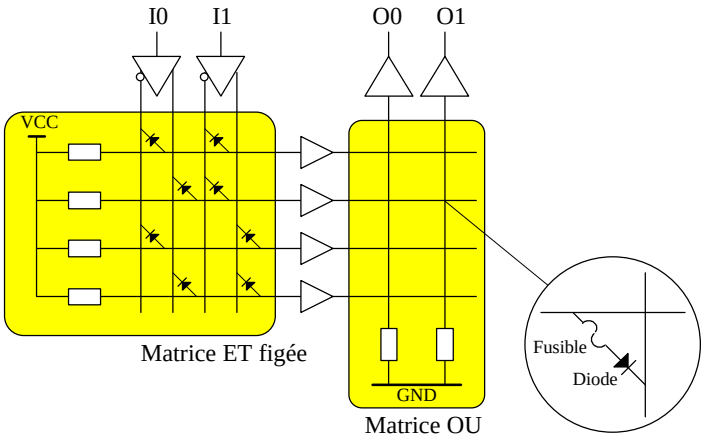
La tendance actuelle pour les circuits logiques programmables sur site est une augmentation de la densité d'intégration permettant à la fois, l'augmentation du nombre de cellules logiques et l'implantation de fonctions logiques complexes d'usage courant prêtes à l'emploi (mémoire RAM, mémoire FIFO, Timers, Diviseurs de fréquence, etc.). Ces nouveaux circuits sont pourvus d'un système de scrutation interne des structures (Boundary Scan, JTAG) pour faciliter leur maintenance par des procédés automatiques.

Circuits logiques programmables chez les fondeurs (ASIC)

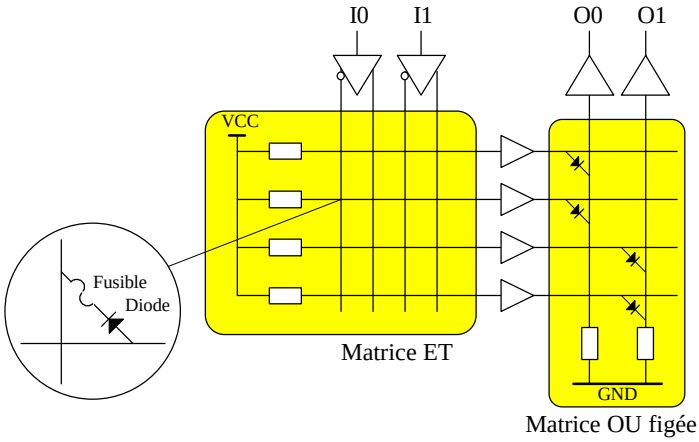
Les fondeurs sont les entreprises qui fabriquent les circuits intégrés. Cette opération nécessite de lourds investissements pour chaque circuit créé. En effet, chaque circuit utilise pour sa fabrication des "masques" de gravure différents, ce qui limite la fabrication des circuits intégrés qu'à des modèles que l'on est sûr de pouvoir vendre en très grande quantité (quelques millions de pièces : circuits standards TTL ou CMOS etc.). L'amélioration des techniques de fabrication et la diminution des coûts associés ont permis de réaliser des circuits intégrés pour des quantités moindres (quelques 100.000 pièces).
Les ASIC sont classables en 3 catégories suivants le degré de personnalisation du composant :

- *réseaux prédiffusés (semi custom)* : Ce sont des réseaux constitués de transistors, portes, cellules non interconnectés entre eux et constituant les premières phases de l'intégration monolithique. L'interconnexion finale (personnalisée) est réalisée par les derniers masques.
- *réseaux pré caractérisés (custom)* : Ce sont des réseaux réalisés suivant toutes les étapes de fabrication d'un circuit intégré. Les fonctions utilisées sont extraites d'un catalogue.
- *circuits à la demande (full custom)* : Comme le nom l'indique, ce sont des circuits qui sont intégralement conçus à la demande du client.

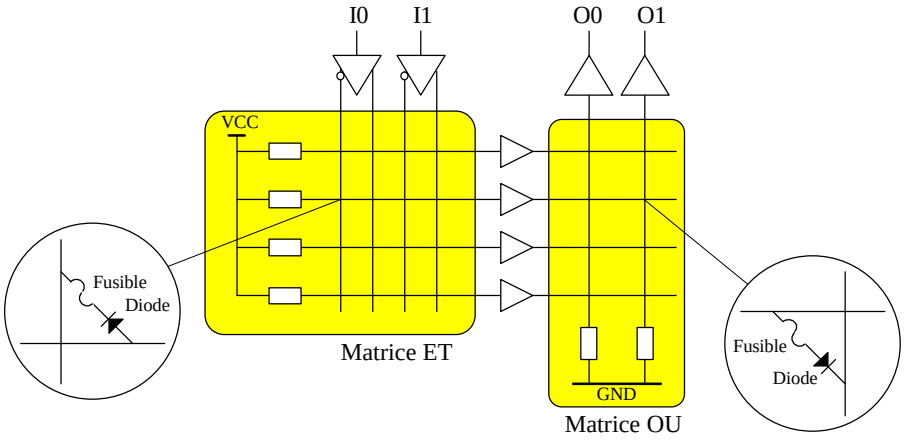
Exemples de réseaux logiques combinatoires programmables à 2 entrées et 2 sorties utilisant des matrices à diodes



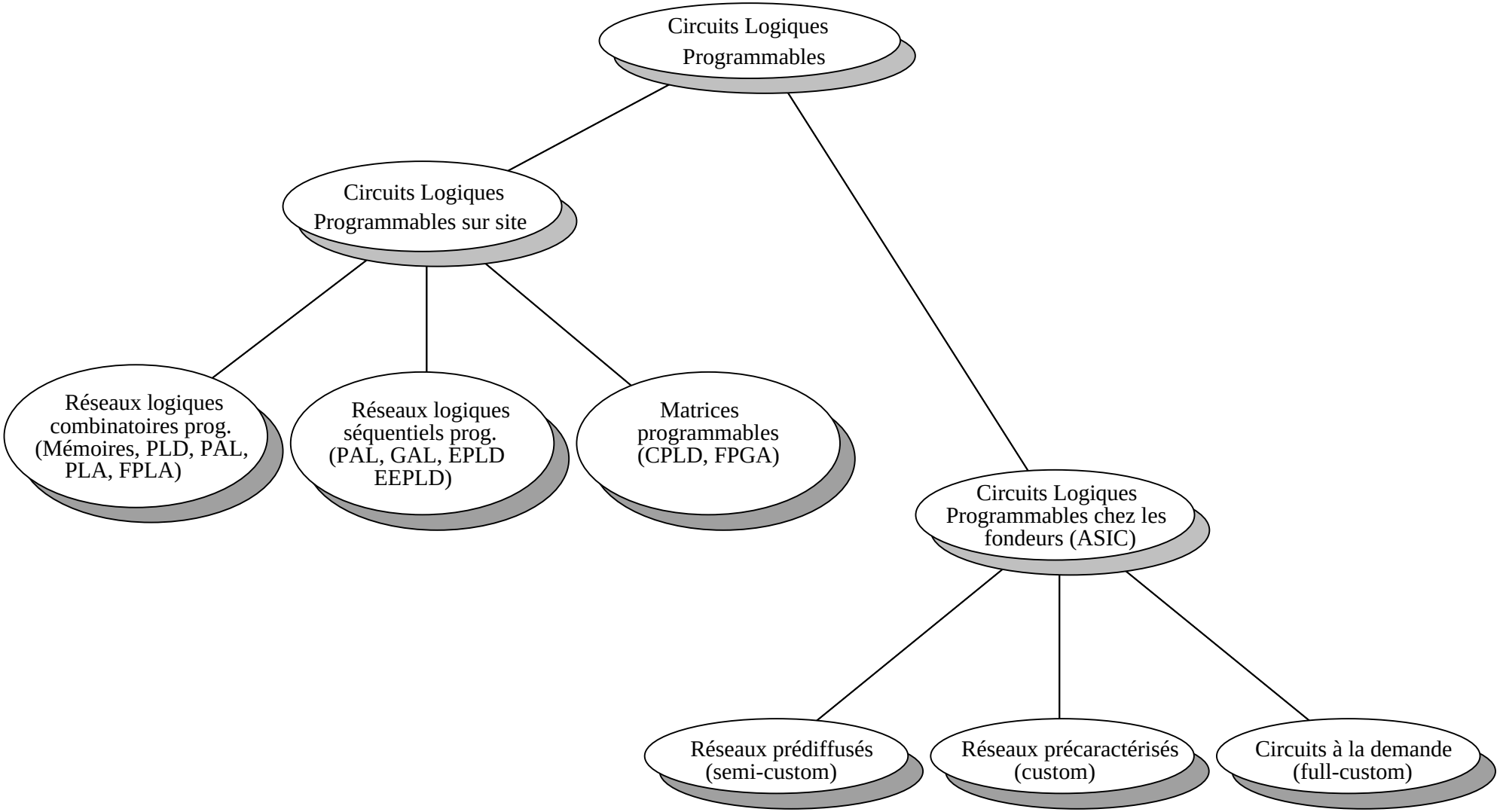
Type d'organisation d'une Mémoire



Type d'organisation d'un circuit PAL



Type d'organisation d'un circuit FPLA


Circuits Logiques Programmables
Circuits Logiques Programmables sur site
Réseaux logiques combinatoires prog. (Mémoires, PLD, PAL, PLA, FPLA)
Réseaux logiques séquentiels prog. (PAL, GAL, EPLD EEPLD)
Matrices programmables (CPLD, FPGA)
Circuits Logiques Programmables chez les fondeurs (ASIC)
Réseaux prédiffusés (semi-custom)
Réseaux précaractérisés (custom)
Circuits à la demande (full-custom)