République Tunisienne

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Direction Générale des Etudes Technologiques

---*---**

Institut Supérieur des Etudes Technologiques de Sousse

Fascicule des travaux pratiques

Circuits programmables

Spécialité : Génie électrique

Elaboré par : TLILI KAIS

TECHNOLOGUE à L'ISET DE SOUSSE

Sommaire

TP1 : PROGRAMMATION DES PLD

Présentation du GAL22V10

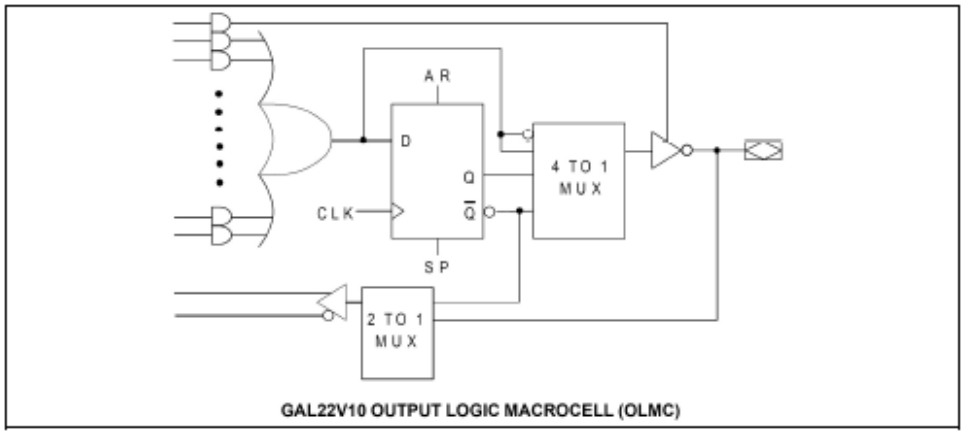
Présentation

Le GAL 22V10 est implanté dans un boitier DIP 24 broches ou PLCC 28 broches. Il appartient à la technologie E2CMOS PLD (E2 = Electrically Erasable). Sa structure interne est donnée à la page suivante.

Ce circuit est doté de 12 entrées dont une à double fonction et de 10 sorties. On remarque que ces 10 sorties sont issues de cellules nommées OLMC (Output Logic Macro Cell). L'analyse de la structure interne (paragraphe suivant) montre que ces sorties sont réinjectables comme entrées.

Structure interne d'une OLMC

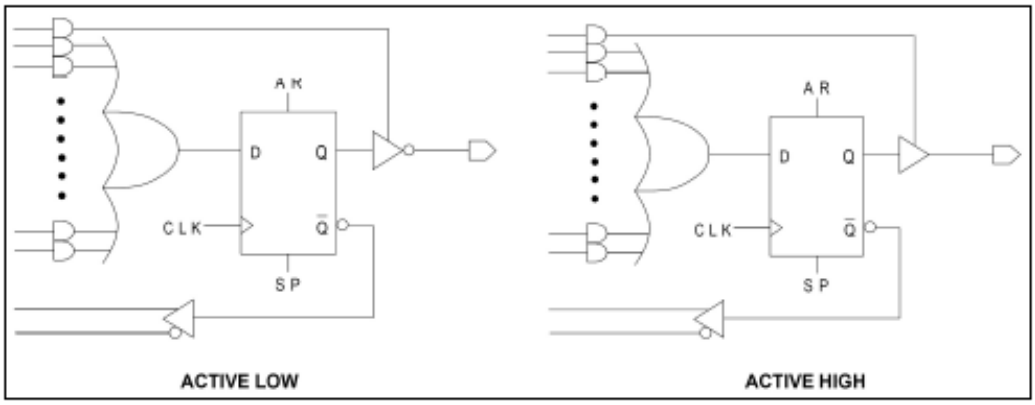
L'OLMC comporte principalement une bascule D et deux multiplexeurs. C'est une structure programmable.



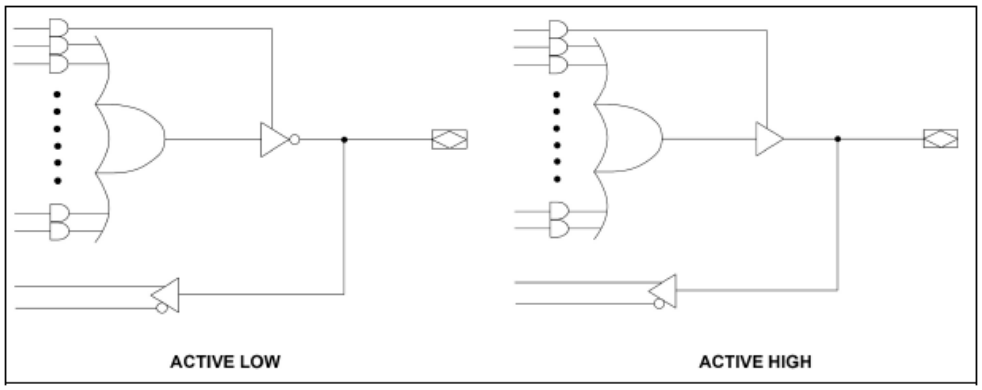
GAL22V10 OUTPUT LOGIC MACROCELL (OLMC)

Les figures données ci-après présentent les principales configurations possibles de l'OLMC.

Mode registre (bascule D utilisée)



Mode combinatoire



Programmation du GAL22V10

Pratiquement on ne programme jamais directement ni les OLMC ni les matrices. On utilise pour cela un logiciel (ISIS de Proteus avec compilateur Ez-ABEL) qui connait la structure interne du composant et qui y réalise l'implantation (fitting) du problème qu'on lui soumet. Néanmoins, la connaissance du composant permet d'évaluer avec efficacité la faisabilité d'un projet.

L'analyse de la structure interne du GAL22V10 met en évidence certaines contraintes d'utilisation et donc de programmation de certaines broches (en mode registered) :

- Les bascules ont une horloge commune (synchrone), broche 1.
- Les bascules sont dotées d'un RESET (mise à 0 des sorties Q) asynchrone et d'un PRESET (mise à 1) synchrone. La programmation du RESET est obligatoire.

La programmation d'un circuit logique programmable s'effectue à partir des équations ou de la table de vérité dans un fichier de description utilisant le langage ABEL.

Etapes de la programmation

- Création d'un projet Proteus et saisie du schéma sous ISIS,
- Création et compilation du fichier ABEL,
- Association du fichier JEDEC obtenu au PLD du schéma,
- Simulation interactive ou avancée (chronogrammes) du système,

Exemple de fichier Abel : dspprog.abl

Module DSPPROG

Title 'alarme de phares'

Declarations

" Référence du composant à programmer

DSPPROG **device 'P22V10';**

" Déclaration des entrées

C, L, P **pin** 2, 3, 4**;**

" Déclaration des sorties

S **pin** 23 **istype 'com';**

Equations

S = !C & L & P**;**

End DSPPROG

Travail demandé

PROGRAMMATION DU GAL22V10

Saisie du schéma

Ouvrir le logiciel ISIS de la suite Proteus.

- Cliquer sur *Fichier -> Nouveau Projet*, choisir le modèle *Landscape A4*.

- Enregistrer votre projet en respectant le nom et le chemin suivants : D:\ez-abel\DSPPROG.DSN

Réalisation du schéma DSPPROG utilisant un PLD 22V10 :

- Cliquer sur l'icône *Mode composant* puis Cliquer sur l'icône *P*,
- Entrer la référence du composant en *Mots clés* (22V10),
- Dans la fenêtre des résultats, choisir *AM22V10* puis valider. Le composant apparaît maintenant dans le sélecteur d'objets à gauche,
- Cliquer sur le schéma pour placer le composant puis réaliser les liaisons du montage : fils d'entrées et sortie correspondant au fichier ABEL.

Labellisation des fils :

- Cliquer sur l'icône *Mode label* puis cliquer sur les liaisons d'entrées et de sortie du montage afin de saisir leurs noms (P, L, C et S).

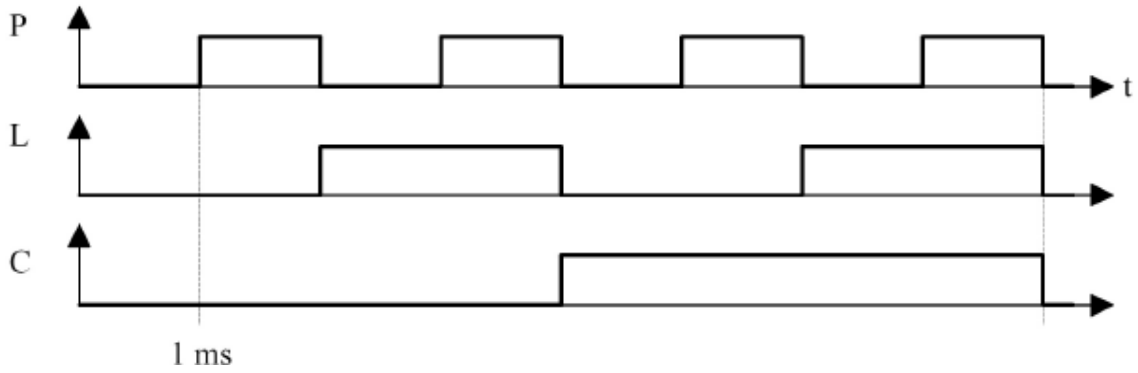
Placement et configuration des signaux d'entrées

Placement des générateurs d'entrées :

- Cliquer sur l'icône *Mode générateur,* sélectionner *DCLOCK* puis cliquer sur les liaisons d'entrées P, L et C du montage. Le générateur doit automatiquement porter le même nom que le label placer précédemment.

Configuration des signaux d'entrées :

- Double cliquer sur chaque générateurs et configurer leurs boites de dialogue afin d'obtenir les signaux ci-dessous :



Placement des sondes de mesures

- Cliquer sur l'icône *Mode sonde de tension* puis cliquer sur la liaison de sortie S du montage.

Placement et configuration des chronogrammes

Placement des chronogrammes sur le schéma :

- Cliquer sur l'icône *Mode graphe.* Sélectionner *DIGITAL* pour effectuer une simulation logique.
- Pour placer le graphe sur le schéma, positionner le curseur dans une zone libre du dessin, enfoncer le bouton gauche de la souris puis faite glisser celle-ci afin d'obtenir un cadre. Relâcher le bouton, un chronogramme vierge apparaît.

Configuration des chronogrammes :

- Double cliquer sur le graphe pour faire apparaître sa boite de dialogue. Ajuster la valeur *Temps fin* qui permettra d'obtenir la durée totale souhaitée.
- Glisser le générateurs et la sonde de tension du schéma dans le graphe afin qu'ils apparaissent dans l'ordre suivant (de haut en bas) : P, L, C et S.

Création et compilation du fichier ABEL

Paramétrage du logiciel :

- Cliquer sur *Source -> Définir outil de génération de code…*, choisir l'outil *ABEL.*

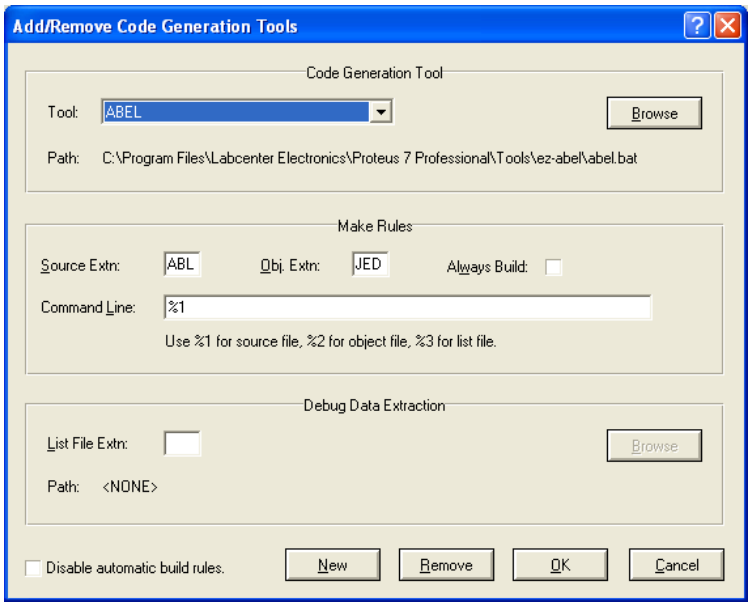
Remarque :

Si l'outil ABEL ne figure pas dans la liste

- cliquer sur *Nouveau* puis sélectionner le fichier :

C:\Program Files\Labcenter Electronics\Proteus 7 Professional\Tools\Ez-abel\Abel.bat

- Configurer l'outil conformément à la figure suivante :



Création du fichier ABEL :

- Cliquer sur *Source -> Ajouter/supprimer fichier source…*, choisir l'outil de génération de code *ABEL*, cliquer sur *Nouveau* et créer le fichier en respectant le nom et le chemin suivants : D:\ez-abel\DSPPROG.ABL
- Cliquer sur *Source -> 1.DSPPROG.abl* afin d'ouvrir l'éditeur de fichier.
- Tapez le code donné à la page 2 et enregistrer les modifications.

Compilation du fichier ABEL :

- Cliquer sur *Source -> Tout construire.* Corriger les erreurs éventuelles dans le fichier ABEL et, le cas échéant, relancer la compilation.

Si vous obtenez le message *COMPILATION EFFECTUEE AVEC SUCCES* alors le fichier JEDEC à bien été créé dans votre projet.

Association du fichier JEDEC obtenu au PLD du schéma

- Effectuer un click droit sur le composant *AM22V10* puis éditer ses propriétés.
- ans le champ *JEDEC Fuse Map File*, parcourir et sélectionner votre fichier JEDEC fraichement créé (D:\ez-abel\DSPPROG.JED).

Lancement de la simulation et exploitation des résultats

Lancement de la simulation :

- Cliquer sur le graphe et appuyer sur la barre d'espace pour lancer la simulation.

Exploitation des résultats :

- Vérifier l'exactitude des chronogrammes afin de valider le fonctionnement du système.

DECODEUR BCD / 7 SEGMENTS

On souhaite intégrer la fonction Décodeur BCD / 7 segments dans un 22V10.

Description du fichier ABEL

On va utiliser dans le cas présent la description par la table de vérité.

Les entrées E1, E2, E3 et E4 seront affectées respectivement aux broches 2, 3, 4 et 5.

Les sorties a, b, c, d, e, f et g seront affectées respectivement aux broches 23 à 17.

TP2 : TP2 : PROGRAMMATION DES PLD EN COMPTEURS ET MACHINE A ETAT

1- OBJECTIFS

Programmer un PLD pour réaliser les fonctions de comptage élémentaires (comptage / décomptage, binaire / décimal et à sorties 7 segments).

2- PRÉSENTATION

Ce travail de programmation comporte deux parties. La première partie constitue un travail d'initiation qui consiste à implanter dans un circuit logique programmable les fonctions de bases réalisées par les compteurs intégrés (comptage binaire ou décimal, décomptage...).

La deuxième partie permettra d'intégrer un compteur et un décodeur BCD / 7 segments pour afficher

directement la valeur de sortie du compteur. La programmation sera réalisée à partir de fichier de

description de type ABEL. Le circuit utilisé est un 22V10.

3- TRAVAIL DEMANDÉ

3.1- RÉALISATION D'UN COMPTEUR BINAIRE MODULO 16

- Créer un Projet **CTRDIV16.DSN** voir (TP1)
- Le fichier de description ABEL qui réalise un compteur binaire modulo 16 est le Suivant :

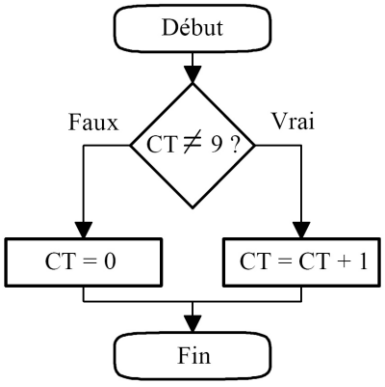
```
Module CTRDIV16 " Nom du module, identique au nom du fichier ABEL
Title 'compteur binaire synchrone modulo 16'
Declarations
CTRDIV16 device 'P22V10';
" Entrées
H, RESET pin 1, 2; " H, l'horloge du compteur, est obligatoirement affectée
" à la broche 1. L'entrée RESET de remise à zéro
" des bascules doit être déclarée.
"Sorties
Q0, Q1, Q2, Q3 pin 23, 22, 21, 20 istype'reg_d';
" Les sorties du compteur sont affectées et déclarées
" comme sorties de bascules D.
CT = [Q3..Q0]; " Les 4 sorties sont définies en un bus nommé CT.
Equations " Description du fonctionnement du circuit.
CT.AR = RESET; " L'entrée de Remise à '0' de chaque bascule
" est reliée à l'entrée RESET.
CT.CLK = H; " L'entrée d'horloge de chaque bascule
" est reliée à l'entrée H.
CT := (CT + 1); " Le compteur est incrémenté sur chaque front actif de H.
" (front montant)
End CTRDIV16
```

- Saisir le schéma relatif au compteur
- Placer les signaux d'entrées et les sondes
- Placer la fenêtre des chronogrammes

3.2- RÉALISATION D'UN COMPTEUR BINAIRE MODULO 10

On veut réaliser un compteur binaire synchrone modulo10.

La réalisation d'un compteur décimal de '0' à '9' s'effectue à partir de la même équation de fonctionnement que celle du compteur modulo16. La seule différence est que le CT du compteur doit être ramené à '0' lorsque la valeur '9' est atteinte.



Ce fonctionnement peut être traduit graphiquement par une structure conditionnelle qui sera exprimée en langage ABEL par la syntaxe :

WHEN (CT !=9) **THEN** CT:= CT+1;

ELSE CT:= 0;

- Créer un Projet **CTRDIV10.DSN** voir (TP1)
- Ecrire un programme ABEL, **CTRDIV10.AB**L qui réalise ce compteur
- Saisir le schéma relatif au compteur
- Placer les signaux d'entrées et les sondes
- Placer la fenêtre des chronogrammes

3.3- COMPTEUR /DECOMPTEUR BINAIRE MODULO 10

On veut réaliser un compteur-décompteur binaire modulo15. Le sens de comptage sera donné par une nouvelle entrée nommée **SENS** et affectée à la **broche 3** du 22V10. Pour la simulation, cette entrée sera à '0' pendant 20ms puis à '1' le reste du temps.

- Créer un Projet **COMDECOM.DSN** voir (TP1)
- Ecrire un programme ABEL, **COMDECOM.ABL** qui réalise ce compteur
- Saisir le schéma relatif au compteur
- Placer les signaux d'entrées et les sondes
- Placer la fenêtre des chronogrammes

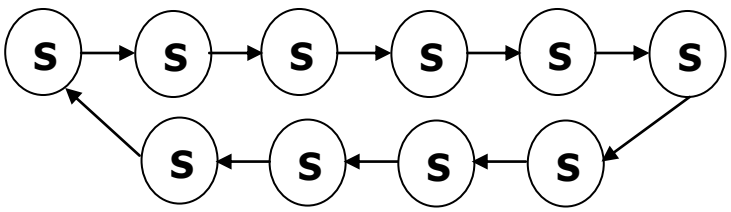
3.4- MACHINE A ETAT

On veut réaliser un compteur décimal qui réalise directement l'affichage du nombre compté sur un afficheur 7 segments (sorties codées en 7 segments au lieu du binaire).

La méthode de description par équations n'est pas adaptée dans ce cas de figure car le passage d'une combinaison de sortie à la suivante ne peut pas être décrit de façon simple. On utilisera ici le **diagramme d'état** (state diagram) :

- Dans la rubrique **declarations** sont définis les différents états (state) de sorties possibles.

- Dans la rubrique **state_diagram** sont définies les conditions de passage d'un état au suivant.



- Compléter le programme suivant

```
Module CTRAF7S
Title'compteur décimal avec sorties codées en 7 segments'
Declarations
CTRAF7S device 'P22V10'; " Nomme le futur fichier JDEC destiné à un 22V10
H, RESET pin 1, 2 ; " Définition des entrées puis des sorties de bascule D
a,b,c,d,e,f,g pin 23, 22, 21,20,19,18,17 istype'reg_d' ;
" déclaration des différents états de sortie possibles
Sraz=[0,0,0,0,0,0,0] ; " état initial (après une remise à 0 : segments éteints)
S0=[1,1,1,1,1,1,0] ; " état des segments pour afficher '0'
S1=[1,1,1,1,1,1,0] ; " à compléter
S2=[0,1,1,0,0,0,0] ; " à compléter
S3=[1,1,1,1,0,0,1] ; " à compléter
S4=[0,1,1,0,0,1,1] ; " à compléter
S5=[1,0,1,1,0,1,1] ; " à compléter
S6=[1,0,1,1,1,1,1] ; " à compléter
S7=[1,1,1,0,0,0,0] ; " à compléter
S8=[1,1,1,1,1,1,1] ; " à compléter
S9=[1,1,1,1,0,1,1] ; " à compléter
Equations
[a,b,c,d,e,f,g].AR = RESET ; " L'entrée R de chaque bascule est reliée à RESET.
[a,b,c,d,e,f,g].CLK = H ; " L'horloge de chaque bascule est reliée à l'entrée H.
State_diagram [a,b,c,d,e,f,g] " Diagramme d'état décrivant l'évolution des sorties
state Sraz : goto S0 ; " Lorsque le compteur se trouve dans l'état Sraz (tous
" les segments éteints), il passe à l'état S0 sur front
" actif de H
state S0 : goto S1;
state S1 : goto S2; " à compléter
state S2 : goto S3; " à compléter
state S3 : goto S4; " à compléter
state S4 : goto S5; " à compléter
state S5 : goto S6; " à compléter
state S6 : goto S7; " à compléter
state S7 : goto S8; " à compléter
state S8 : goto S9; " à compléter
state S9 : goto S0; " à compléter
end CTRAF7S
```

3.5- COMPTEUR /DECOMPTEUR DECIMAL AVEC SORTIE 7 SEGMENTS

L'entrée SENS, affectée à la broche 3, réalisera la commande de comptagedécomptage :

- Si SENS = '0' : décomptage.
- Si SENS = '1' : comptage.

La réalisation du compteur-décompteur décimal de '0' à '9' avec affichage 7 segments s'effectue à partir du même fichier que celui utilisé précédemment. La différence de fonctionnement est traduite dans le diagramme d'état où le **passage d'un état au suivant sera conditionné** par la variable SENS.

Cette structure sera exprimée en langage ABEL par la syntaxe :

state N **: IF** SENS = = 1 **THEN** N+1 **ELSE** N-1;

Le diagramme d'état du fichier ABEL sera donc modifier comme suit :

State_diagram [a,b,c,d,e,f,g] // diagramme d'état décrivant l'évolution des sorties

state Sraz : goto S0 ;

state S0 : if (SENS==1) then goto S1 else S9;

state S1 : if (SENS==1) then goto S2 else S0; …

TP3 : MACHINE A ETAT FINI

Objectifs

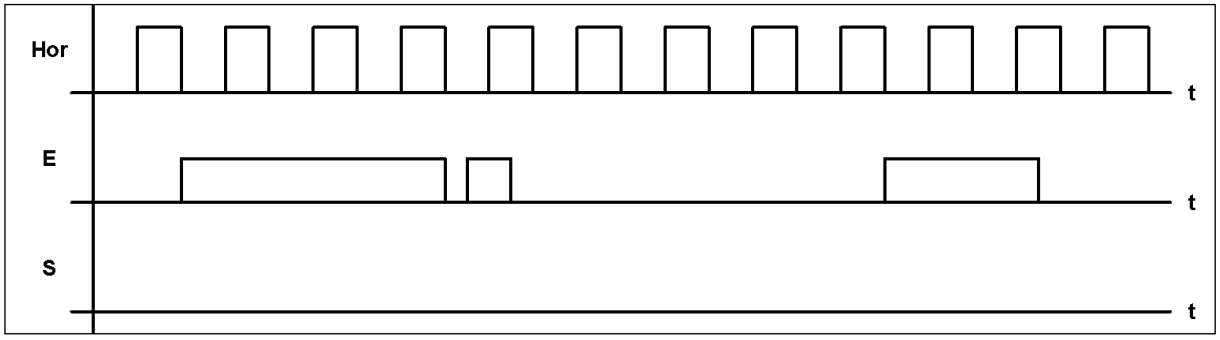
- ⇨ Résoudre des problèmes séquentiels en utilisant les machines à état fini.
- ⇨ Valider le fonctionnement des machines à état par simulation autour d'un GAL22V10

Travail demandé

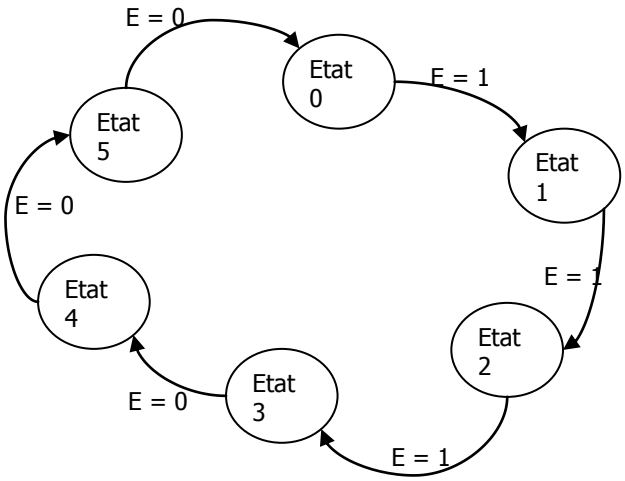
1. Réalisation d'une bascule T à partir d'une bascule D.
   i. Donner la table de vérité de la bascule T.
   ii. Représenter le diagramme de transition en utilisant la machine de Moore
   iii. Déterminer à partir du digramme de transition l'équation de la sortie Q.
   iv. Programmer et simuler sous ISIS la bascule T en langage ABEL (il est demandé de tester les 3 méthodes : equations, truth_table et state_diagram).
2. Refaire le même travail pour une bascule JK.

3. On désire réaliser une fonction dont la sortie S recopie l'état présent sur son entrée E, si celle-ci est restée stable après 2 coups d'horloge consécutifs.

i. Tracer la sortie S sur le chronogramme suivant :



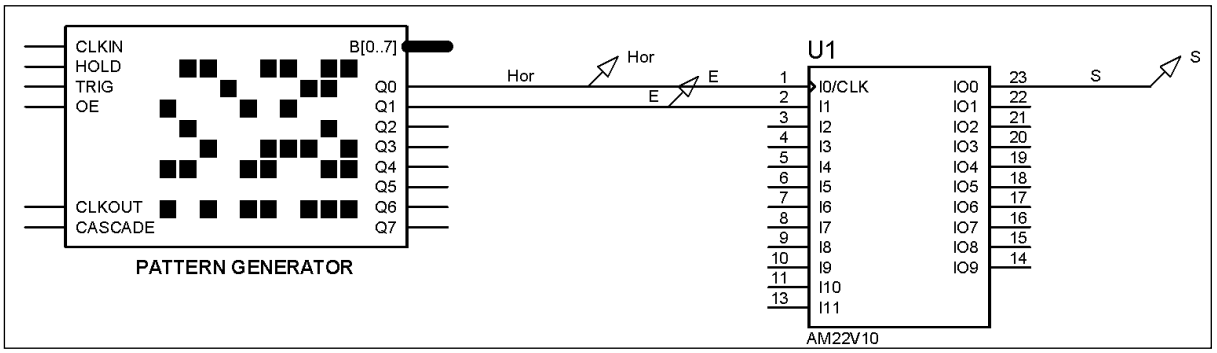
ii. Compléter le diagramme de transition



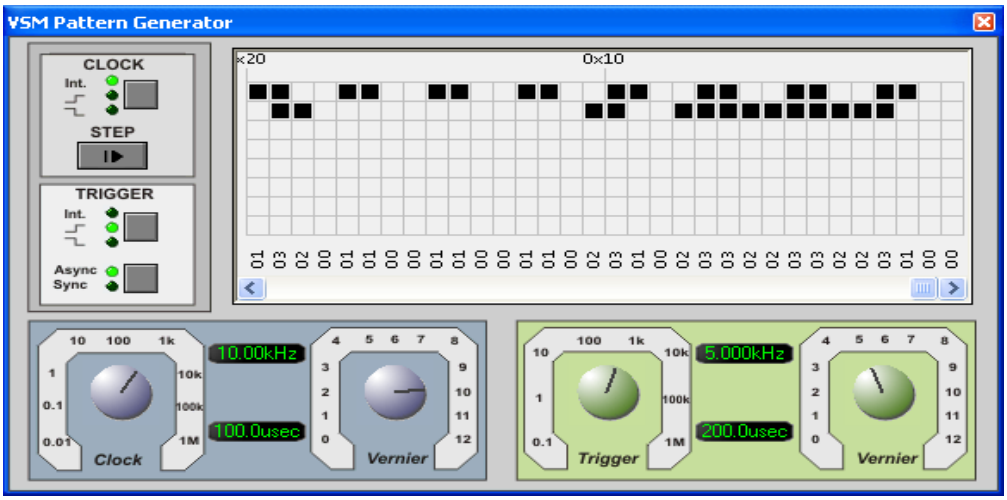
1. Traduire le digramme de transition en langage ABEL

⇨ Simulation sous ISIS :

- Saisir sous ISIS le schéma suivant



- Cliquer sur le bouton « Pause » pour faire apparaitre la fenêtre du générateur de trame virtuel « PATTERN GENERATOR ». Faite la configuration comme indiqué dans la fenêtre suivante pour générer les signaux **Hor** et **E** sur les sorties Q0 et Q1.



- Visualiser les signaux **Hor**, **E** et **S** en utilisant le « Graph Mode ».

TP4 : TP4 : PROGRAMMATION DES PLD EN MONOSTABLE

4- OBJECTIFS

Programmer un PLD pour réaliser les fonctions monostables et générations des trains d'impulsions.

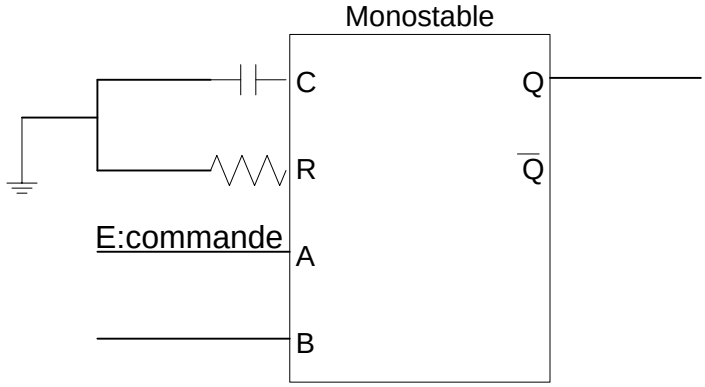
5- PRÉSENTATION

Ce travail de programmation comporte deux parties. La première partie constitue la réalisation d'une fonction monostable qui consiste à l'implanter dans un circuit logique programmable.
La deuxième partie permettra d'intégrer une fonction pour générer un trin d'impulsion.
La programmation sera réalisée à partir de fichier de description de type ABEL. Le circuit utilisé est un 22V10.

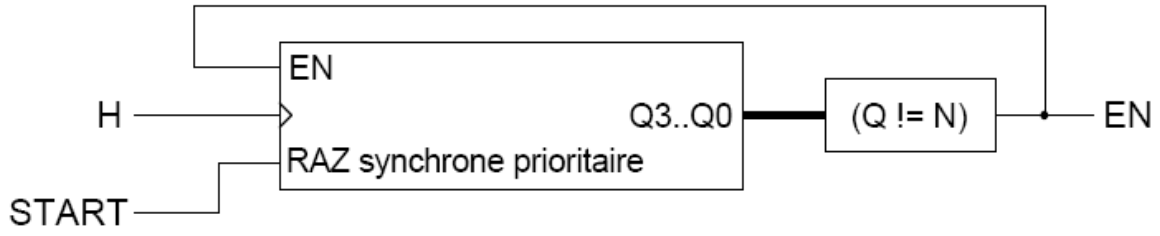
6- TRAVAIL DEMANDÉ

3.1- RÉALISATION D'UN E FONCTION MONOSTABLE

En électronique le monostable délivre en sortie une impulsion calibrée par un circuit RC. Cette impulsion est déclenchée par un chargement de niveau de l'entrée du monostable.



Cette même fonction monostable peut être réalisée par programmation, La durée caractéristique de la fonction monostable est générée par un comptage binaire jusqu'à une valeur prédéfinie.



START : declenchement du monostable
EN : sortie du monostable et autorisation de comptage

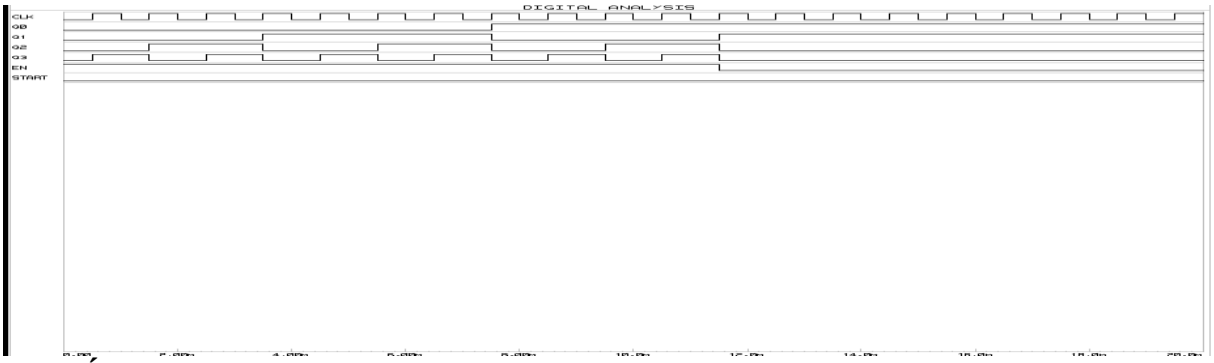
- Créer un Projet **MONOS.DSN** voir (TP1)
- Le fichier de description ABEL **MONOS.ABL** qui réalise une fonction monostable, en prenant les conditions suivantes :

- START =1 alors Q=0
- EN=1 alors incrémentation de Q

➢ Saisir le schéma relatif au montage

➢ Placer les signaux d'entrées et les sondes

➢ Placer la fenêtre des chronogrammes en prenant H=1KHz et T =12ms



3.2- RÉALISATION D'UN E FONCTION TRAIN DE N IMPULSIONS

Il s'agit d'une variante du précédent monostable. Celui-ci est non redéclenchable et l'on utilise le bit de poids faible du compteur pour générer les impulsions. Le nombre d'impulsions générées est N/2 +1.

➢ Créer un Projet **TRAIN.DSN** voir (TP1)

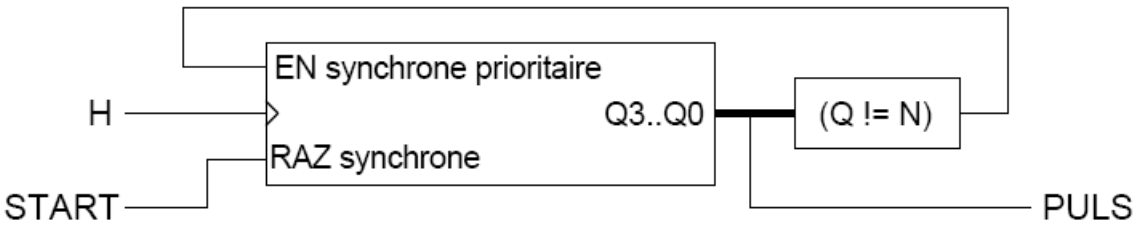
➢ Le fichier de description ABEL **TRAIN.ABL** qui réalise une fonction trains de N impulsions, en prenant les conditions suivantes :

- START =1 alors Q=0
- EN=1 alors incrémentation de Q

➢ Saisir le schéma relatif au montage

➢ Placer les signaux d'entrées et les sondes

➢ Placer la fenêtre des chronogrammes en prenant H=1KHz et T =12ms



START : declenchement du train d'impulsion
EN : Validation du comptage
PULS : sortie du train d'impulsion