RÉALISATION DE COMPTEURS PAR COMPOSANT LOGIQUE PROGRAMMABLE

A• OBJECTIFS

- Programmer un PLD pour réaliser les fonctions de comptage élémentaires (comptage / décomptage, binaire / décimal et à sorties 7 segments).
- Vérifier expérimentalement la validité des programmations.

B• DOCUMENTS NÉCESSAIRES

- Manuel d'utilisation de ViewLogic (Programmation d'un circuit logique prog.).
- Cours : 'Circuits logiques programmables'.
- TP N°4 : ' Initiation à la programmation des PLD'.

C• MATÉRIEL UTILISÉ

- Ordinateur équipé de ViewLogic.
- Imprimante.
- Maquette pédagogique 22V10 et programmateur.

D• 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 base réalisées par les compteurs intégrés (comptage binaire ou décimal, décomptage...).

La deuxième partie permettra d'intégrer un compteur et un décodeur BCD / 7segments 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 le PALCE22V10H-25PC/4 produit par AMD.

E• TRAVAIL DEMANDÉ

I• RÉALISATION D'UN COMPTEUR BINAIRE MODULO 16

1• CONFIGURATION DU PROJET (PAGE 36)

Ouvrir **votre** projet de travail. Les librairies suivantes doivent impérativement être présentes ou ajoutées :

- F:\VENDOR\DIO\DIO(DIO)
- F: \LOGIQUE\BUILTIN (BUILTIN)
- F:\VENDOR\DIO\DSTD (DSTD)

2• DESCRIPTION DU FICHIER ABEL

Le fichier de description ABEL qui réalise un compteur binaire modulo 16 est le suivant :

Module CTRDIV16 // _____

Title 'compteur binaire synchrone modulo 16'

" Entrées
H, RESET pin 1, 2; // _____

// _____

// _____

"Sorties // _____

Q0, Q1, Q2, Q3 pin 23, 22, 21, 20 istype 'reg_d';

// _____

// _____

CT = [Q3..Q0]; // _____

Equations // _____

CT.AR = RESET; // _____

// _____

CT.CLK = H; // _____

// _____

CT := (CT + 1); // _____

// _____

End CTRDIV16

3• ÉCRITURE DU FICHIER ABEL : INTELLIFLOW (PAGE 39)

Lancer IntelliFLow. Ouvrir le fichier CTRDIV16.ABL
Compléter le fichier selon l'exemple donné ci-dessus.

4• CHOIX DU CIRCUIT : INTELLIFLOW (PAGE 40)

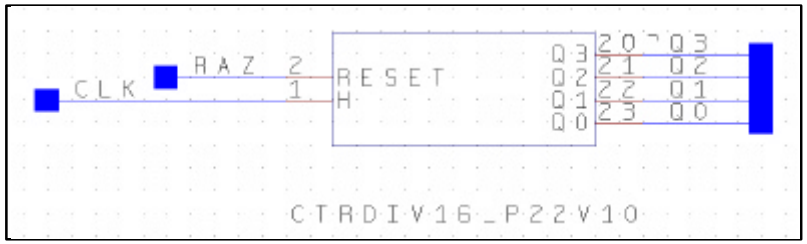
La définition des critères de choix est la même que celle utilisée lors du TP d'initiation (TP N°4). Dans l'onglet **PLD Device Datasheet,** où apparaissent tous les composants répondant aux critères définis, sélectionner le **PALCE 22V10H-25PC/4** de AMD.

5• CRÉATION DU FICHIER JEDEC : INTELLIFLOW (PAGE 41)

Créer le fichier JEDEC (CTRDIV16.JED) ainsi que le modèle de simulation.

6• SIMULATION LOGIQUE DU COMPOSANT CRÉE

Lancer Viewdraw et ouvrir un nouveau fichier de dessin : COMPTE16
Insérer le composant créé (CTRDIV16.1) dans le schéma.
Penser à changer les propriétés du composant en module (à la place de composite).



Ecrire le fichier de commande permettant de vérifier le fonctionnement :
- Définir une impulsion initiale à '1' sur RAZ d'une durée de 0.2ms.
- Définir un signal d'horloge CLK de période 2ms.
- Visualiser les signaux RAZ, CLK, Q0, Q1, Q2 Q3 et la valeur décimale de sortie du compteur.

La durée de la simulation est fixée à 20 périodes du signal d'horloge.
Visualiser les chronogrammes et valider le fonctionnement du montage.

7• VÉRIFICATION EXPÉRIMENTALE (AVEC UN CIRCUIT PROGRAMMÉ)

Tester le fonctionnement du composant programmé avec la maquette pédagogique :
- Alimenter la carte entre 0 et +5V.
- Connecter la sortie TTL du GBF à l'entrée d'horloge H. La fréquence sera réglée à 2 Hz environ.

Valider le fonctionnement de la maquette. **Faire vérifier par le professeur.**

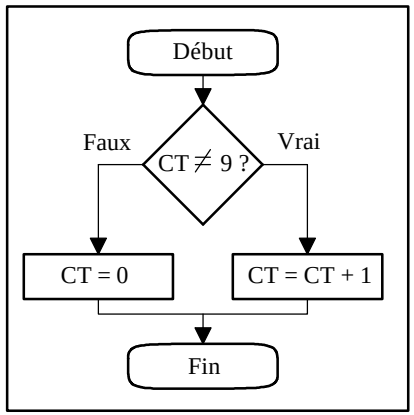
Quelle est l'action de l'interrupteur RESET ? Cette entrée est-elle prioritaire sur le fonctionnement du compteur ?

II• COMPTEUR BINAIRE MODULO 10

On veut réaliser un compteur binaire synchrone modulo10.

1• ÉCRITURE DU FICHIER ABEL

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 :



Cette structure sera exprimée en langage ABEL par la syntaxe :

WHEN (CT !=9) **THEN** CT:= CT+1;
ELSE CT:= 0;

Ce qui peut se traduire par :

QUAND CT ≠ 9 **ALORS** CT = CT + 1
SINON CT = 0

FICHIER CTRDIV10.ABL

MODULE _____

TITLE _____

// Entrées

_____

// Sorties

_____

_____

EQUATIONS

_____

_____

_____

_____

END CTRDIV10

Ecrire le fichier ABEL CTRDIV10 (à partir du fichier précédent) en remplaçant l'équation par celle donnée ci dessus. **Attention au respect des parenthèses et des signes de ponctuation !**

2• SIMULATION LOGIQUE DU COMPOSANT CRÉE

Reprendre les différentes étapes de la création du composant. Effectuer la simulation du composant (fichier dessin : COMPTE10) et valider le fonctionnement du montage.

3• VÉRIFICATION EXPÉRIMENTALE (AVEC UN CIRCUIT PROGRAMMÉ)

Tester le fonctionnement du composant programmé avec la maquette pédagogique. **Faire vérifier par le professeur.**

III• COMPTEUR DÉCOMPTEUR BINAIRE MODULO 10

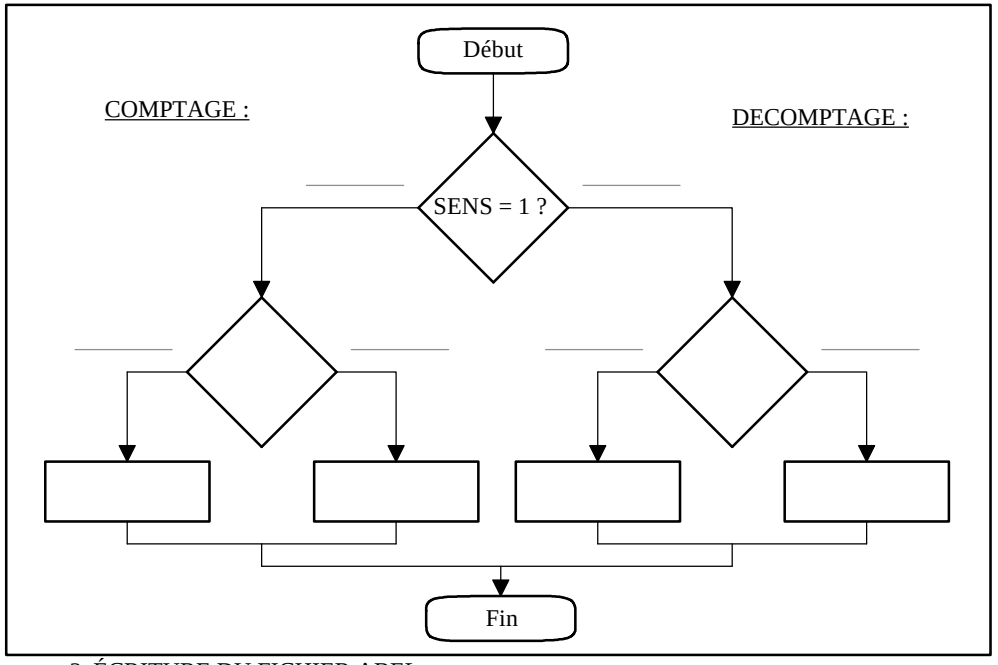
On veut réaliser un compteur / décompteur binaire synchrone modulo10.

Le circuit est doté d'une entrée supplémentaire : SENS (broche 3). Cette entrée réalise la commande de comptage / décomptage :

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

1• ALGORIGRAMME DE DESCRIPTION

En vous aidant de l'exemple précédent et des infos ci-dessus, complétez l'alorigramme afin de décrire le fonctionnement du compteur / décompteur :



2• ÉCRITURE DU FICHIER ABEL

En vous aidant de l'exemple précédent, compléter le fichier ABEL pour obtenir un compteur / décompteur décimal. Ecrire le fichier ABEL pour réaliser cette fonction :

FICHIER CTRDEC.ABL

MODULE _____

TITLE _____

// Entrées

_____

// Sorties

_____

_____

EQUATIONS

_____

_____

WHEN (SENS = = 1) THEN {WHEN ( ) THEN ;

ELSE ;}

ELSE {WHEN ( ) THEN ;

ELSE ;}

END CTRDEC

3• SIMULATION LOGIQUE DU COMPOSANT CRÉE

Reprendre les différentes étapes de la création du composant. Effectuer la simulation du composant (fichier dessin : DECOMP10) et valider le fonctionnement du montage.

4• VÉRIFICATION EXPÉRIMENTALE (AVEC UN CIRCUIT PROGRAMMÉ)

Tester le fonctionnement du composant programmé avec la maquette pédagogique. **Faire vérifier par le professeur.**

IV• RÉALISATION D'UN COMPTEUR DÉCIMAL AVEC SORTIES CODÉES EN 7 SEGMENTS

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.

1• DESCRIPTION DU FICHIER ABEL

Le fichier de description ABEL CTRAF7S.ABL qui réalise le compteur compteur décimal avec sorties 7 segments est **donné incomplet.**

Module CTRAF7S

Title 'compteur décimal avec sorties décodées 7 segments'

H, RESET pin 1, 2 ; // Définition des entrées

a,b,c,d,e,f,g pin 23, 22, 21,20,19,18,17 istype 'dc,reg_d' ;
// Chaque sortie est définie comme une sortie de bascule D

Equations

[a,b,c,d,e,f,g].AR = RESET ; // L'entrée de R de chaque bascule est reliée à l'entrée RESET.

[a,b,c,d,e,f,g].CLK = H ; // L'entrée d'horloge de chaque bascule est reliée à l'entrée H.

@dcstate // Cette directive associée à l'extension 'dc' permet de
// s'affranchir des combinaisons de sortie non spécifiées

Declarations // déclaration des différents états (state) 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=_____ **// à compléter**

S2=_____ **// à compléter**

S3=_____ **// à compléter**

S4=_____ **// à compléter**

S5=_____ **// à compléter**

S6=_____ **// à compléter**

S7=_____ **// à compléter**

S8=_____ **// à compléter**

S9=_____ **// à compléter**

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
// segments éteints), **il passe à l'état S0** sur front actif de H

state S0 : goto S1;

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

end CTRAF7S

2• ÉCRITURE DU FICHIER ABEL : INTELLIFLOW (PAGE 39)

Ouvrir le fichier CTRAF7S.ABL. Compléter le fichier ci-dessus.
Ne pas oublier de définir :

- Dans la rubrique declarations **tous les états de sortie possibles** (pour afficher les chiffres entre '0' et '9').
- Dans la rubrique state_diagram **la succession de tous les états de sortie**.

3• SIMULATION LOGIQUE DU COMPOSANT CRÉE

Reprendre les différentes étapes de la création du composant.

Ecrire le fichier de commande permettant de vérifier le fonctionnement :
- Définir les signaux d'entrée (horloge et remise à '0')
- Visualiser les entrées et les sorties de commande des segments a, b, c, d, e ,f et g.

Effectuer la simulation (fichier dessin : COMPTAFF) du composant et valider le fonctionnement du montage.

4• VÉRIFICATION EXPÉRIMENTALE (AVEC UN CIRCUIT PROGRAMMÉ)

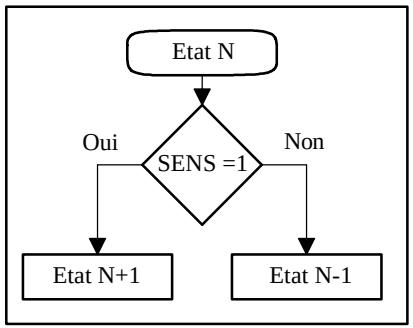
Tester le fonctionnement du composant programmé avec la maquette pédagogique. **Faire vérifier par le professeur.**

V• RÉALISATION D'UN COMPTEUR DÉCOMPTEUR PAR 10 AVEC SORTIES CODÉES EN 7 SEGMENTS

1• ÉCRITURE DU FICHIER ABEL

L'entrée SENS, affectée à la broche 3, réalisera la commande de comptage / décomptage :
- 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. Ce fonctionnement se traduit graphiquement par une structure conditionnelle :



Cette structure sera exprimée en langage ABEL par la syntaxe :

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

Ce qui peut se traduire par :
SI SENS vaut '1' **ALORS** aller à l'état N+1
SINON aller à l'état N-1

Ecrire le fichier ABEL CTRDEC7S en reprenant le fichier précédent et complétant le diagramme d'état :

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

2• SIMULATION LOGIQUE DU COMPOSANT CRÉE

Reprendre les différentes étapes de la création du composant. Effectuer la simulation du composant (fichier dessin : DECOMTAF) et valider le fonctionnement du montage.

3• VÉRIFICATION EXPÉRIMENTALE (AVEC UN CIRCUIT PROGRAMMÉ)

Tester le fonctionnement du composant programmé avec la maquette pédagogique. **Faire vérifier par le professeur.**