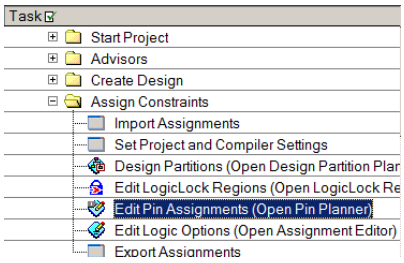
VHDL TP2

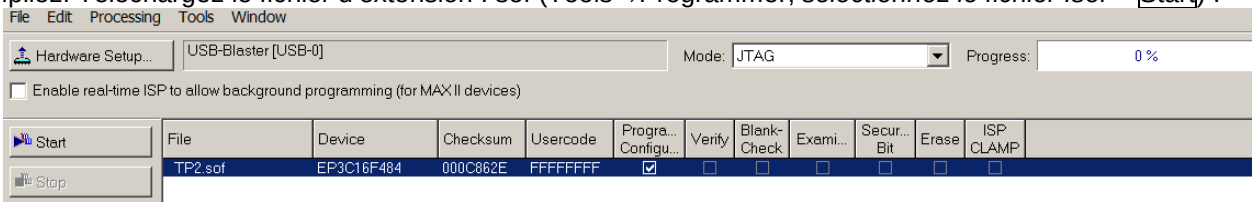
1. Pins Assignement.

Ouvrez votre projet Encodeur.qpf. Pour l'instant il compile et sa simulation est correcte.
Lors de la création du projet, vous avez bien indiqué le composant cible (Cyclone 3, EP3C16F484C6). C'est bien celui qui est présent sur la carte DE0. Les Pin de ce composant sont reliées aux éléments de la carte (Switch, Leds, BP, Afficheurs 7 segments…). Il ne reste plus qu'à indiquer vers quelle PIN du composant doivent être mappés les signaux de votre description VHDL.

En vous aidant de la documentation *DE0_User_manual.pdf* sur *U:\Documents\DUT\GEII\ModulesS3\ESE-VHDL* pages 24, 25 et 26, et en sachant qu'on veut utiliser les interrupteurs Slide Switch[0], Slide Switch[1], Slide Switch[2] et Slide Switch[3] et les LED Green[0], LED Green[1] et LED Green[2] pour les signaux de l'entité encodeur, complétez le Pin Planner :

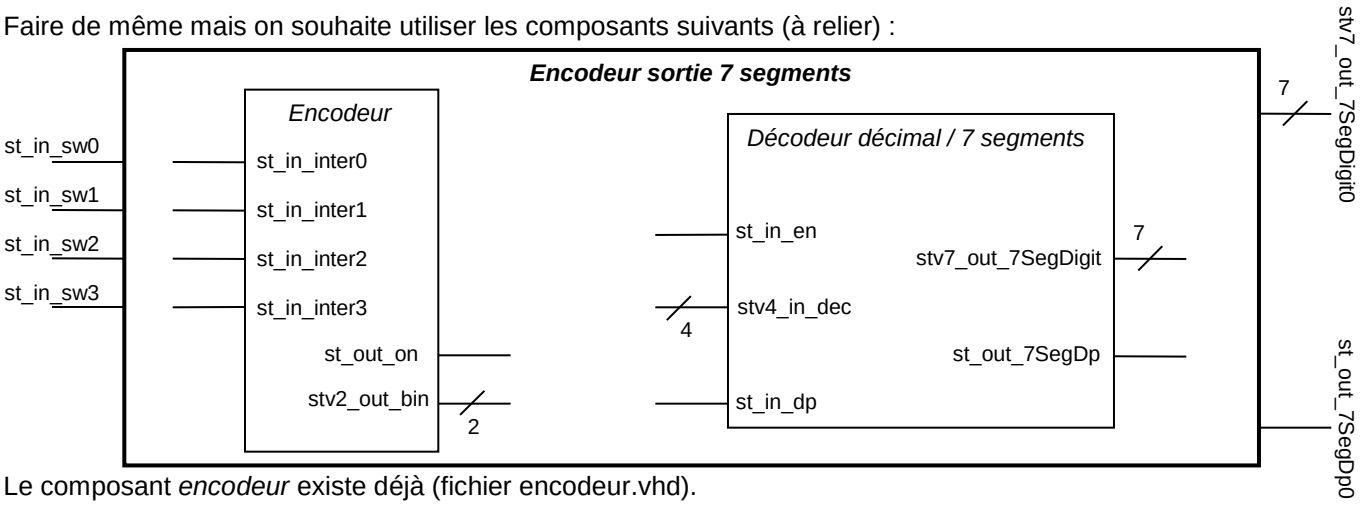


Compilez. Téléchargez le fichier d'extension . sof (Tools→Programmer, *sélectionnez le fichier .sof* + Start) :



2. Avec un afficheur 7 segments .

Faire de même mais on souhaite utiliser les composants suivants (à relier) :



Le composant *encodeur* existe déjà (fichier encodeur.vhd).

Le composant *Décodeur décimal / 7 segments* est à créer (fichier decod7seg.vhd). Il doit afficher le bon chiffre en fonction de stv4_in_dec sur un afficheur 7 segments (voir page 26 de DE0_User_manual.pdf) :

stv4_in_dec	st_in_dp	st_in_en	Affichage
X	X	H	Rien
0 1 2 3 4 5 6 7 8 9	L	L	0 1 2 3 4 5 6 7 8 9
10 11 12 13 14 15	L	L	A b C d E F
X	H	L	Chiffre de stv4_in_dec + Décimal Point

Vue la doc, les LED des afficheurs 7 segments sont-elles à cathodes communes ou à anodes communes ?

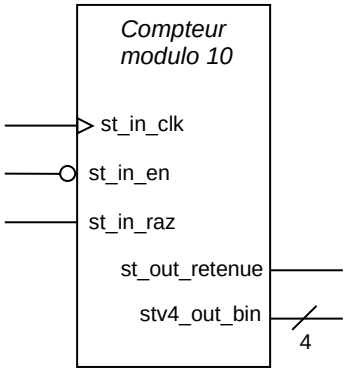
Le composant ***Encodeur sortie 7 segments*** est une instanciation des 2 précédents. Le chiffre affiché sur l'afficheur 7 segment 0 de la carte doit correspondre au numéro de l'interrupteur actif (sw0 à sw3).
Rien n'est affiché si aucun interrupteur n'est actif. Le Décimal Point doit être éteint en permanence.

On demande 4 fichiers : encodeur.vhd, decod7seg.vhd, package_Encod7seg.vhd (qui contient la définition du package) et Encod7seg.vhd (qui contient les instanciations et les liaisons entre les signaux des composants instanciés de l'entité hiérarchique supérieure).

3. Chronomètre avec un graphe d'état.

3.1 Avec Quartus, créer un nouveau projet Chrono (pensez à d'abord créer un dossier Chrono / Voir TP1).

3.2 Créez et testez en simulation ce composant :

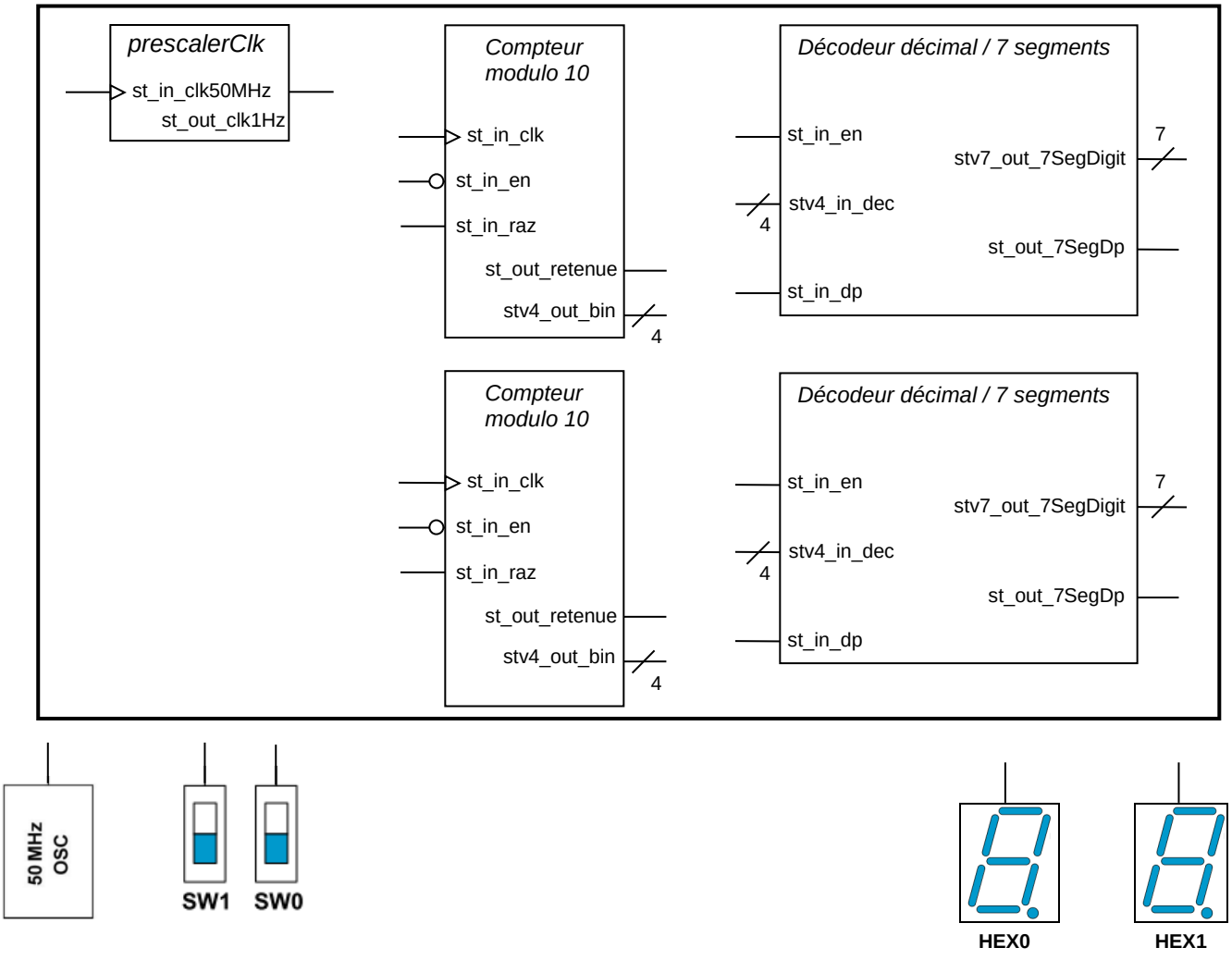


De la plus grande priorité à la plus faible :
- Remise à 0 (st_in_raz).
- Fige le comptage (st_in_en)
- Compte les fronts montants de l'horloge st_in_clk.

st_out_retenue passe à 1 lors du dernier état de comptage.

3.3 Dans le même projet, on souhaite disposer d'un composant (prescalerClk) permettant d'obtenir une horloge de fréquence 1Hz à partir de celle de la carte (DE0_User_manual.pdf pages 28 29). Testez ce composant en simulation.

3.4 Dans le dossier de votre projet actuel, coller le fichier decod7seg.vhd de l'exercice 2. Ajoutez-le à votre projet (project Navigtor, Files d → Add/Remove Files in Project). Créez et testez en réel l'application suivante (faites les liaisons avant de coder). Cette application doit compter et afficher (afficheurs 7 segments 0 et 1) les secondes écoulées (de 0 à 99). On doit avoir la possibilité de figer ce comptage et de le remettre à 0 via les interrupteurs SW0 et SW1.



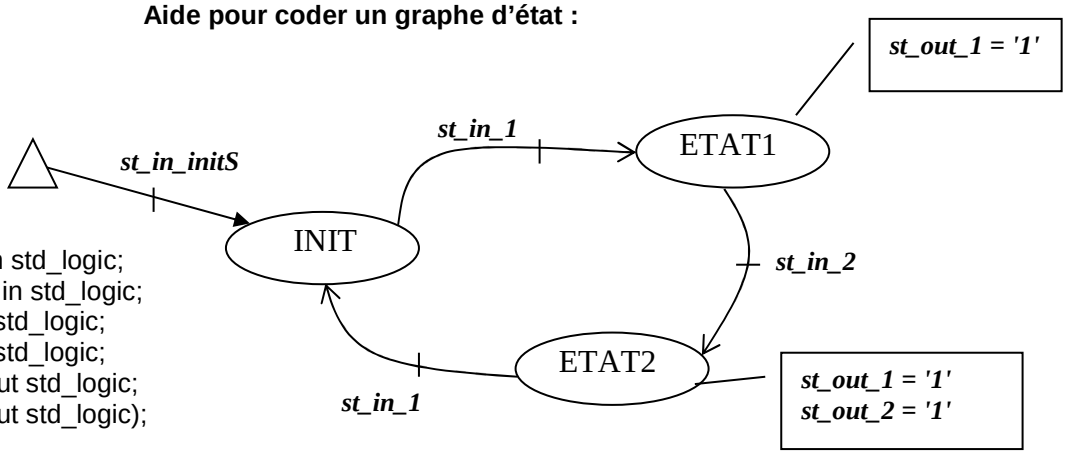
Améliorez en utilisant les 4 afficheurs 7 segments (ajoutez 10ème et 100ème de seconde).

3.5 Cahier des charges du chronomètre.
On souhaite réaliser un chronomètre :

- Lors du premier appui sur le bouton poussoir Button2, le chronomètre doit de mettre en fonctionnement et défier sur les afficheurs 7 segments.
- Lors d'un second appui sur le bouton poussoir Button2, le chronomètre doit se figer.
- Lors d'un 3ème appui sur le bouton poussoir Button2, le chronomètre doit se remettre à 0.

Donner le graphe d'état correspondant à ce fonctionnement
Modifiez la modélisation de votre application en ajoutant un composant nommé *automate* qui code ce graphe d'état et pilotes les signaux de contrôle des Compteurs. Sw0 permet de remettre le graphe en état initial.

Aide pour coder un graphe d'état :




```
entity automate is port(
        signal st_in_clk : in std_logic;
        signal st_in_initS : in std_logic;
        signal st_in_1 : in std_logic;
        signal st_in_2 : in std_logic;
        signal st_out_1 : out std_logic;
        signal st_out_2 : out std_logic);
end automate;

Architecture automate_arch of automate is
        type TEtat is ( INIT, ETAT1, ETAT2);                    -- définition du type énum TEtat
        signal etat : TEtat;                                    -- déclaration du signal Etat
begin
        process(st_in_clk)                                      --évolution de l'état : process
        begin
                if(st_in_clk'event and st_in_clk = '1') then
                        if (st_in_initS='1') then               --initialisation du graphe
                                etat <= INIT;
                        else
                                case etat is                    -- description du graphe
                                        when INIT =>            -- on utilise une instruction case
                                                if (st_in_1 = '1') then   --(équivalente au switch en C)
                                                        etat <= ETAT1;
                                                end if ;
                                        when ETAT1=>
                                                if (st_in_2 = '1') then
                                                        etat <= ETAT2;
                                                end if ;
                                        when ETAT2=>
                                                if (st_in_1 = '1') then
                                                        etat <= INIT;
                                                end if ;
                                end case;
                        end if ;
                end if;
        end process;

        st_out_1 <= '1' when((etat = ETAT1) or (etat = ETAT2))  -- calcul des sorties,
                else '0';                                       -- affectations simultanées
        st_out_2 <= '1' when(etat = ETAT2)
                else '0';

end automate_arch;
```