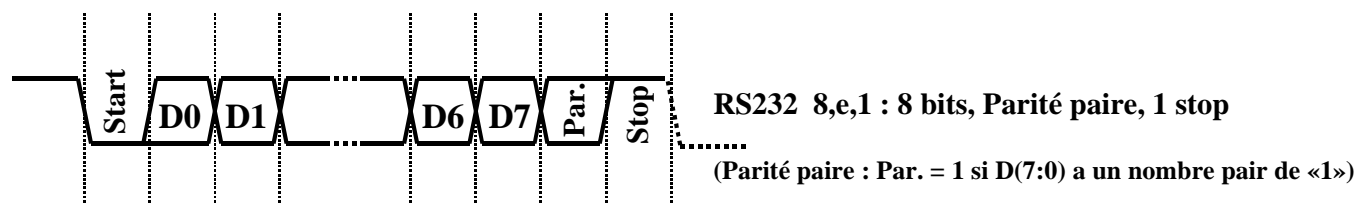


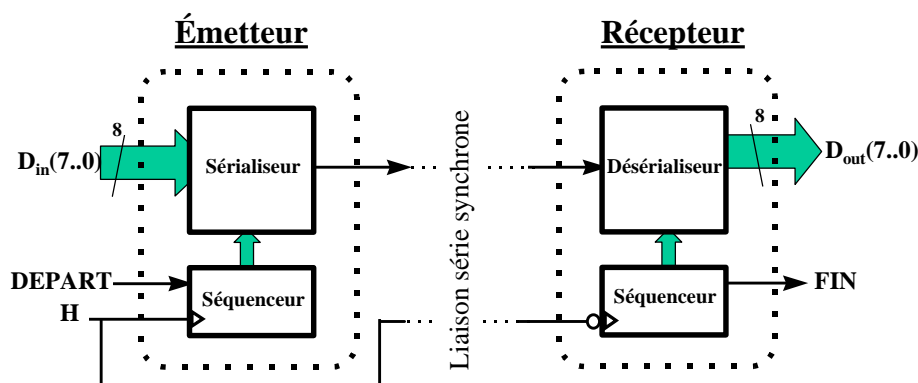
Travaux de Laboratoire

"Electronique des Systèmes Logiques"

L'objectif de cette série de TP est la réalisation d'un système d'émission / réception série synchrone qui s'inspire du protocole RS232 dans sa version 8 bits de données, parité paire, 1 bit de stop.



Le système est donc constitué d'un module d'émission (EM) et d'un module de réception (REC). Chacun d'entre eux est constitué de deux blocs : un séquenceur et une unité associée au signal; la liaison entre ces blocs comprend l'horloge et la liaison série.



Le signal FIN permet de prévenir le circuit placé en aval de la réception d'un nouveau message ; ce signal ne doit être validé que si les données ont été correctement réceptionnées (contrôle de parité et de présence de STOP).

Le module émetteur doit être synthétisé par une approche de multiplexage, celui-ci ne doit être opéré que sur les 8 bits de données; il faudra donc utiliser un multiplexeur 8 bits (8->1); la génération des signaux de START et STOP doit se faire en logique combinatoire classique (portes logiques).

Le module récepteur doit fournir des signaux (D_{out} et FIN) compatibles avec les signaux d'entrée de l'émetteur (sorties D_{out} stables entre 2 informations de FIN).

La conception doit être faite avec un souci **d'optimisation du nombre de ressources utilisées** ; à titre d'exemple : la synthèse des modules de séquençement (machine à 11 états) ne doit utiliser, comme éléments séquentiels, que 4 bascules (ou un compteur 4 bits) et pas plus.

Les quatre séances qui constituent ce TP sont organisées comme suit :

Première séance

mise au point du séquenceur du module EM, prise en main d'Altera

Seconde séance

mise au point du module EM complet

Troisième séance

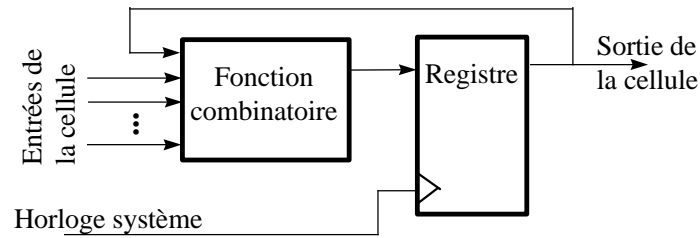
mise au point du système complet et compilation temporelle

Quatrième séance

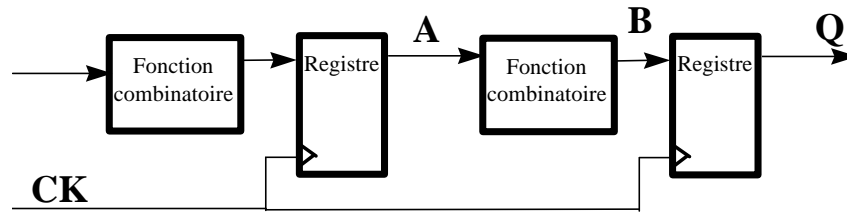
Programmation du composant, analyse temporelle, test sur plaquette

Le système de transmission sera réalisé en logique **synchrone**.

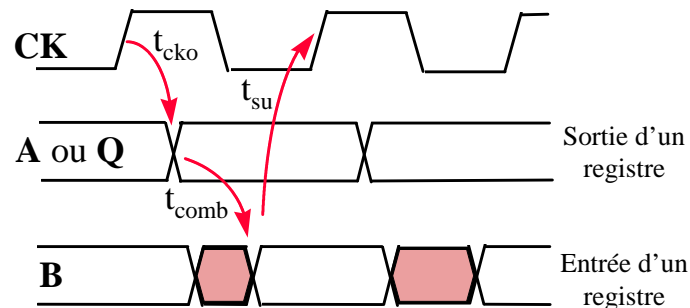
La cellule de base dans une telle logique séquentielle est la suivante :



Le système sera donc composé de plusieurs cellules mises en cascade :



L'horloge à laquelle est associée chacun des registres permet d'échantillonner les données A qui deviennent alors stable après un temps t_{cko} . Ces données sont alors prises en compte dans une fonction combinatoire dont la sortie B est stable après un temps t_{comb} .



Pour que le flux des données soit correctement traité, il faut donc que la sortie B soit stable et respecte le temps de *setup* du registre qui va l'échantillonner. Il faut donc que :

$$T_{ck} > t_{cko} + t_{comb} + t_{su}$$

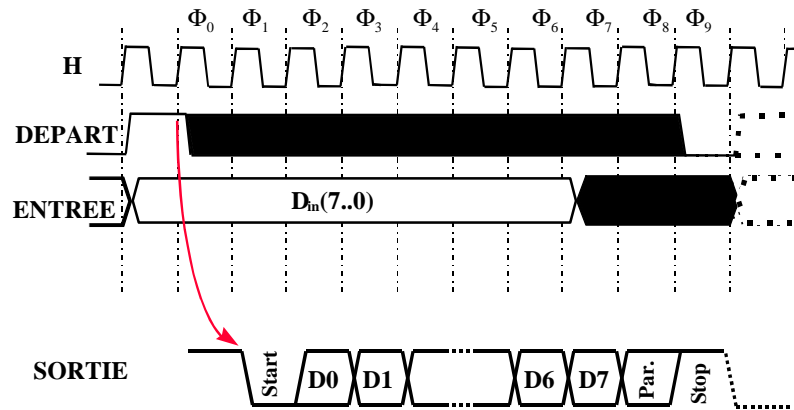
Nota :

1. l'horloge (globale à un module) ne doit être employée que sur les entrées d'horloge des bascules
2. Il est **STRICTEMENT** interdit d'utiliser les entrées asynchrones des bascules et compteurs (*Set/Reset*) pour effectuer une fonction logique.

Séance 1

Au cours de cette première séance, vous devez mettre au point le séquenceur qui permet de contrôler le flux des données d'entrée.

Ce module gère en entrée les signaux d'horloge (**H**) et de départ (**DEPART**) et délivre en sortie un signal qui permet au module spécifiquement lié aux données, de les exploiter correctement. L'unité complète EM génère alors le signal **SORTIE** qui sera transmis.



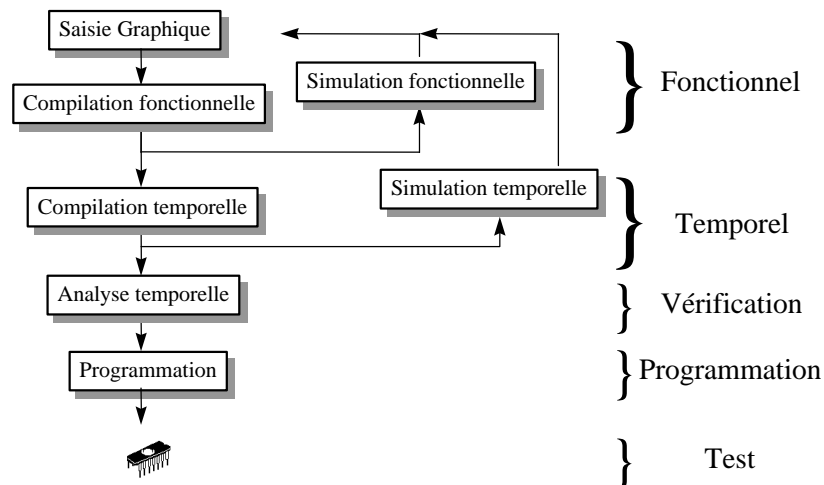
Vous remarquerez que l'on échantillonne le signal **DEPART** et que l'on ne reconsidère sa valeur que lorsque la donnée à sérialiser a été complètement transmise.

Si le signal '**DEPART**' est à l'état bas lorsqu'il est évalué par le sérialiseur, on est en état d'attente, on ne transmet alors que des bits à '1' (état de **STOP**).

Peu importe le temps de latence entre le moment où le signal **DEPART** est à l'état haut et l'instant où le premier bit de la donnée **D** est transmis.

Le logiciel Max+2 vous permet de saisir graphiquement votre séquenceur et d'en faire une simulation fonctionnelle.

La chaîne de conception est la suivante :



Vous aborderez l'étape fonctionnelle au cours des deux premières séances.

Les quatre dernières phases de conception seront mise en œuvre au cours des séances 3 et 4.

Saisissez graphiquement votre séquenceur (*Graphic Editor*), compilez-le (*Compiler*) et vérifiez à l'aide de l'éditeur de signaux (*Waveform Editor*) et du simulateur (*Simulator*) la validité des signaux de contrôle destinés au sérialiseur.

Lorsque vous avez activé le *Waveform Editor*, Activez le menu *Nodes/Inter nodes from SNF/List* pour forcer les signaux d'entrée aux valeurs désirées.

Faites transiter les signaux sur une phase inactive de l'horloge (le front descendant par exemple), afin de lever l'ambiguïté sur l'instant d'échantillonnage.

Vous utiliserez un pas de simulation d'une micro seconde, et une durée de simulation de 100µs.

Séance 2

Mise au point du module EM complet.

Le sérialiseur que vous devez concevoir sera constitué d'un multiplexeur. Il aura la charge d'aiguiller les données d'entrées en fonction des signaux issus du séquenceur.

Faites-en une saisie graphique et simulez-le fonctionnellement, l'objectif étant de respecter le protocole de la ligne série RS232.

Séance 3

Mise au point du système complet.

Vous remarquerez que le récepteur exploite l'inverse du signal d'horloge envoyé aux deux modules, et qu'il génère un signal indiquant la fin de la transmission.

La fonctionnalité du système doit être maintenue lors du passage de la compilation fonctionnelle vers la compilation temporelle, sans avoir à modifier votre conception.

Faites en sorte que l'ensemble de la chaîne émission/réception occupe le moins de place possible dans le composant (Altera FLEX10K20).

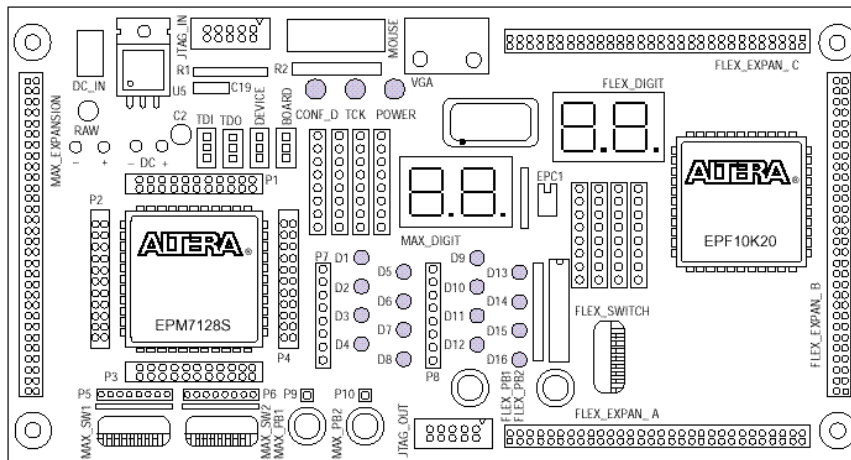
Projetez votre conception sur ce composant (*Assign/Device/Flex10K/Flex10k20RC240-4*), réalisez une compilation dans le but de faire une analyse *temporelle* (*Menu Compiler : Processing/Timing SNF extractor*), et visualisez les signaux de sorties.

La fréquence d'horloge sur la ligne de transmission doit être supérieure à 25 MHz. Vous prendrez soin en particulier de ne pas avoir de *glitches* sur la ligne de transmission série.

Le signal FIN doit pouvoir être interprété comme le signal DEPART vis à vis d'un autre module de transmission série.

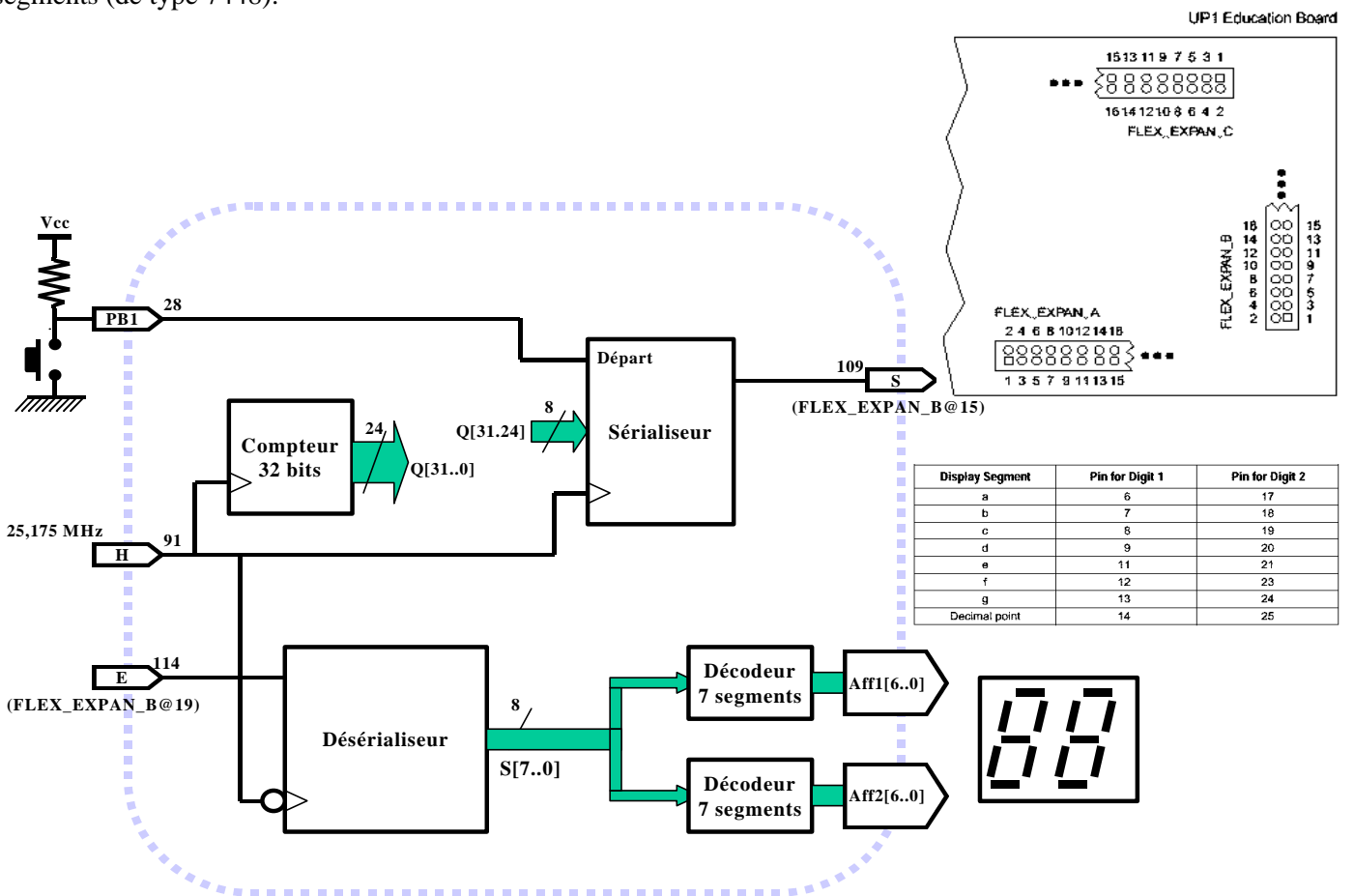
Séance 4

Vérifiez que votre conception respecte bien les contraintes imposées fonctionnelle et temporelle imposées. Le montage sera testé sur une carte de développement (Altera UP1), présentée ci-dessous.



Carte ALTERA UP1

Le montage de test proposé est présenté par la figure suivante, l'idée est d'afficher l'état d'un compteur cadencé à 1,5 Hz sur 2 afficheurs 7-segments (FLEX_DIGIT) ; pour réaliser ce montage, il faut placer un compteur de 32 bits dont les 8 bits de poids fort seront transmis. L'affichage sera fera par le biais de 2 décodeurs BCD-7-segments (de type 7448).



Montage de test sur carte ALTERA UP1

Le compteur et le décodeur d'affichage se trouvent sur :

<http://www.supelec-rennes.fr/ren/rd/etscm/base/altera/up1/util/>

Il est également prévu de pouvoir interrompre la transmission lorsque l'on appuie sur le bouton FLEX_PB1.

Il vous faut forcer le compilateur de Max+2 à répartir les signaux d'entrée/sortie sur des broches spécifiées sur la figure précédente. Pour ce faire, activez le '*Floorplan Editor*', cliquez dans *Layout / Current Assignment Floorplan* et répartissez les '*Unassigned nodes*' sur les broches choisies.

Lancez la compilation, puis vérifiez dans le '*Floorplan Editor*' lorsque l'option dans *Layout / Last Compilation Floorplan* est activée, que le compilateur a bien respecté vos contraintes.

Afin de faire une simulation générale, il vous sera utile d'indiquer au simulateur que les broches d'entrée/sortie de la ligne série sont reliées à l'extérieur du composant. Cela sera réalisé en actionnant l'option *Assign/Connected pins* et en déclarant comme appartenant au même groupe les deux broches considérées.

Testez ensuite votre composant à l'aide de l'oscilloscope et de l'analyseur logique.

Quelques commandes MaxPlus II

Note : le logiciel gère les données sous forme de PROJET (dont le nom est indiqué dans le bandeau supérieur de la fenêtre, aussi il est IMPORTANT de veiller, lors d'une compilation ou d'une simulation, à ce qu'il s'agisse bien de celui que l'on veut traiter ; la définition du projet se fait par le menu : *File/Project/ ...*

Les éléments logiques dont vous aurez besoin se trouvent dans 2 bibliothèques, PRIM (fonctions primitives) et MF (macrofonctions).

Graphic Editor

Symboles

Ajout :	<i>Symbol/Add_Symbol</i> ou <i>Double_Click</i>
Utiles :	<i>Vcc, Gnd.....</i> Forçage de niveau <i>Input, Output.....</i> E/S de la feuille <i>NOT, AND2, NAND3, etc... ..</i> Portes logiques de base (bibliothèque PRIM)
Conseillés :	<i>dff</i> Bascule D <i>enadff.....</i> Bascule D avec validation <i>81mux.....</i> Multiplexeur 8->1 <i>barrelst.....</i> Registre à Décalage (voir table de vérité !) <i>4count</i> Compteur 4bits (voir table de vérité !)
Fonctionnalité :	<i>Help/Macrofunctions</i> ou bouton avec "?"
Schéma :	<i>Double_Click</i>
Taille de la feuille	<i>File/Size/Automatic.....</i> H ou V
Affichage de la grille	<i>Options/Show_Guidelines</i> Non (recommandé)
Collage des lignes	<i>Options/Rubberbanding</i> Oui (recommandé)

Compiler

Compilation fonctionnelle :	<i>Processing/Functionnal.....</i> Oui, pour les premières séances
Compilation temporelle :	<i>Processing/Functionnal.....</i> Non, pour les dernières séances
Choix de composant :	<i>Assign/Device/Flex10K/Flex10k20RC240-4</i> Pour les dernières séances

Waveform Editor

Durée de simulation :	<i>File/End_Time</i> 100us
Pas de la grille :	
Définition :	<i>Options/grid_size.....</i> 1us
Accrochage :	<i>Options/Snap_to_Grid.....</i> Oui
Signaux :	
Ajout (après compil.) :	<i>Node/Enter_Node_from_SNF</i> puis <i>List</i>
Modification :	Icones dans la barre de gauche