I) Introduction

Nous allons partir d'une version corrigée de l'ATTiny861 avec rs232 et en faire un certain nombre de modifications. Dans cette épreuve , on ne vous demande pas de vous poser des questions sur l'utilité ou non des modifications mais seulement de les exécuter et de montrer que le travail résultant fonctionne toujours. Le but de l'épreuve est ainsi d'évaluer comment vous appréhendez des modifications à partir d'un schéma complexe décrit en VHDL.
Vous disposez d'une feuille réponse sur laquelle vous répondez et sur laquelle l'enseignant notera ce qu'il a vu fonctionner et sinon, quelques indications sur l'état d'avancement de votre travail.
Une ressource "Resources2015.zip" vous est fournie (dans le répertoire TPCmptPassage).
Le barème n'est donné qu'à titre indicatif.

II) ATTiny861, compteur BCD et RS232

La partie matérielle vous est complètement donnée dans le répertoire TPCmptPassage. Elle correspond à la correction du TP9 (figure ci-dessous). En voici un schéma fonctionnel.

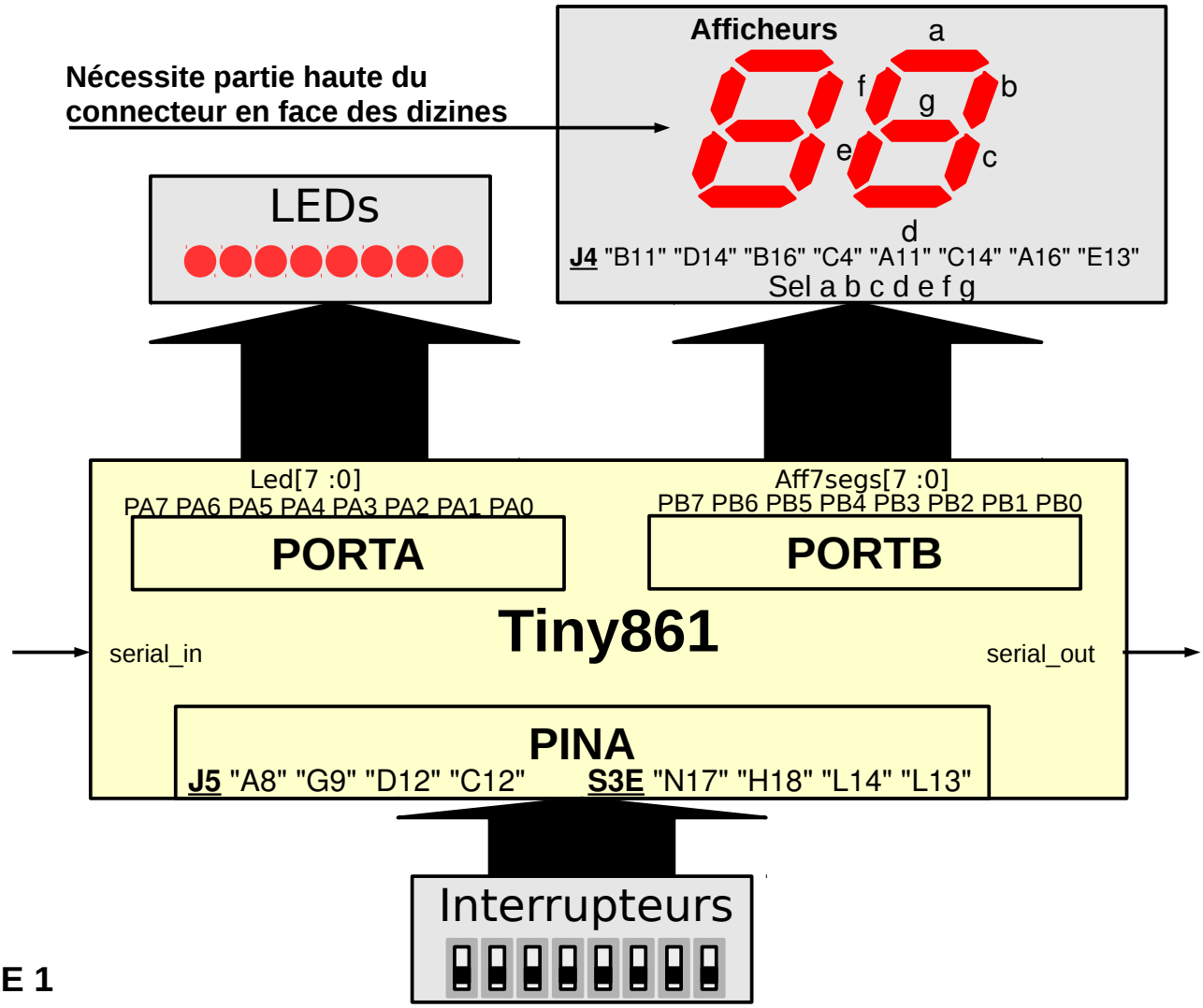

FIGURE 1

Travail à réaliser (3 pts exo1)

2-1) On donne un programme qui réalise un compteur de passages sur deux digits. Le faire fonctionner. Comment modifier le programme pour qu'il affiche correctement ?

INFO

Ce travail consiste donc à :
- faire un projet (Xilinx IDE)
- décompacter la ressource dans le répertoire du projet Xilinx !!!!!!
- ajouter tous les fichiers (sauf cmpPassageUTT.vhd) et compiler le projet (Xilinx IDE)
- compiler le programme C (script dans « soft » en modifiant peut-être le nom du programme à compiler)
- télécharger l'ensemble dans le FPGA (Impact)

Travail à réaliser (6 pts exo2)

On vous demande de déporter la partie logicielle de transcodage dans le matériel.

2-2) Réaliser cette modification matérielle conformément au schéma partiel ci-dessous :

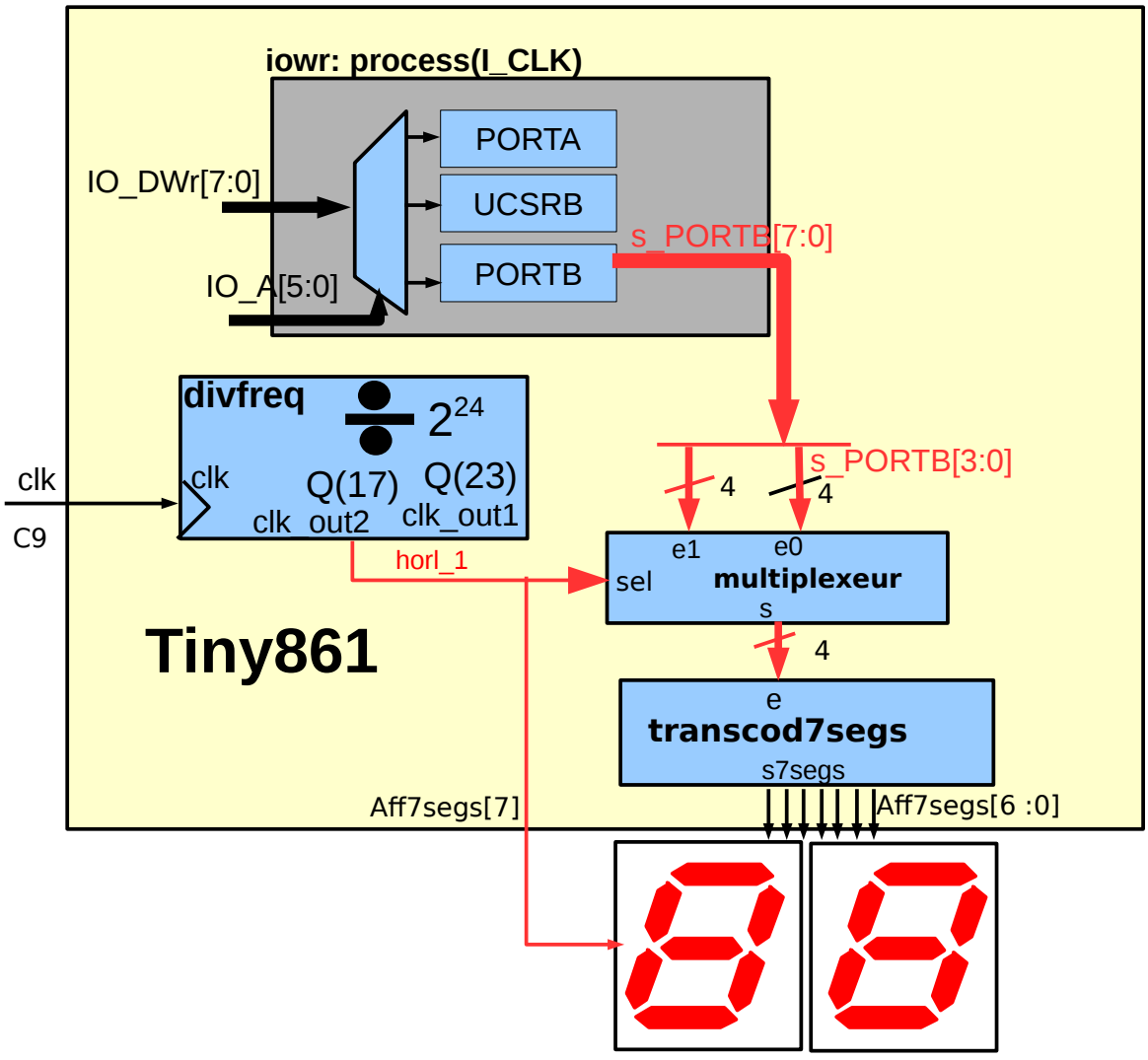

FIGURE 2

INFO

Les rectangles bleus de la figure peuvent être trouvé dans le fichier « cmpPassageUTT.vhd » **PORTB** ne sort plus vers l'extérieur mais est relié au multiplexeur. L'écriture de 0x97 dans le PORTB affichera « 97 »

Ce qui est à faire est en rouge.

2-3) Modifier le programme correspondant pour que l'ensemble continue à fonctionner. Vous devrez retirer complètement le transcodage en C et la gestion des deux afficheurs.

On désire maintenant déporter cet affichage sur un écran VGA.

III) Préparation de l'écran VGA en mode texte

Le principe de base est réalisé par le schéma ci-dessous :

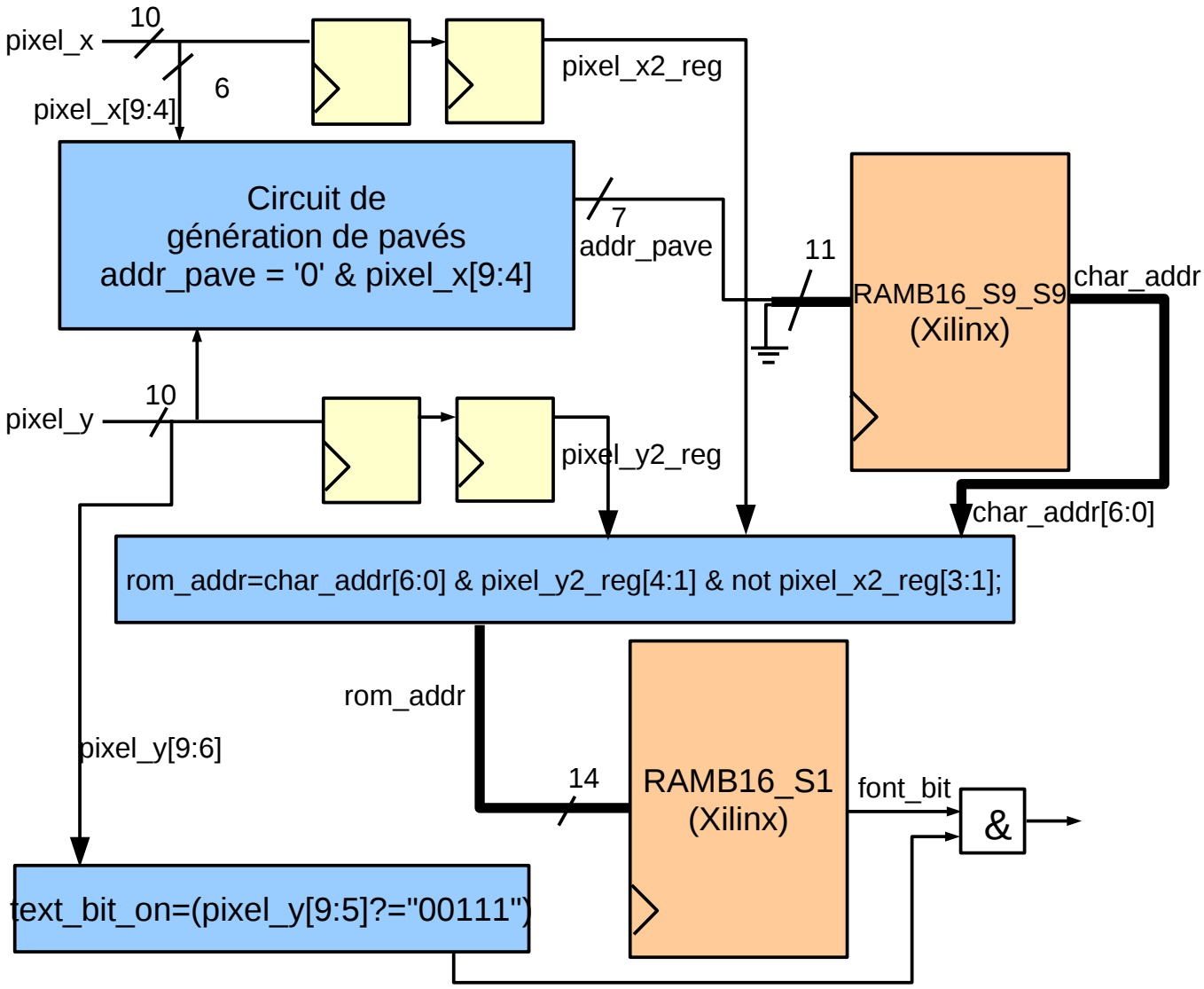

Figure 3

Pas de panique cet ensemble vous est donné (dans VGAStart.vhd) et était même présent dès le premier exercice ! (à votre insu en somme et câblé très partiellement)
Le point essentiel est l'apparition de deux mémoires :
- une ROM de caractères (RAMB16_S1) dans laquelle on stocke la forme de ces caractères,
- une RAM vidéo (RAMB16_S9_S9) dans laquelle on stocke ce qui sera dessiné sur l'écran.

Travail à réaliser (3 pts exo3)

3-1) Vous ne l'aviez sans doute pas remarqué, mais la figure 1 est incomplète. La sortie VGA est gérée et

affiche quelque chose sur l'écran VGA. Qu'est-ce qui est affiché sur l'écran ?

Suivant la configuration de la salle, il est possible que seul l'écran de votre PC puisse être branché sur votre carte FPGA. Prenez soin alors de mettre toutes vos applications sur l'autre écran.

Puis remplir la RAMB16_S9_S9 pour que soit affiché :
"UTT TROYES-Final 2015-Passages : 00"
au milieu de l'écran en lieu et place de l'ancien texte. Les lettres minuscules (particulièrement le 'g') n'étant pas esthétiquement jolies vous pouvez les remplacer par des majuscules.

Vous pouvez afficher la chaîne de caractère que vous voulez, mais il doit y avoir quelque part deux '0' où l'on affichera le nombre de passages.

ATTENTION : la RAMB16_S9_S9 s'initialise à partir de la droite.

Travail à réaliser (5 pts exo4)

3-2) On vous demande de réaliser l'interfaçage avec l'**ATTiny861** comme indiqué ci-dessous.

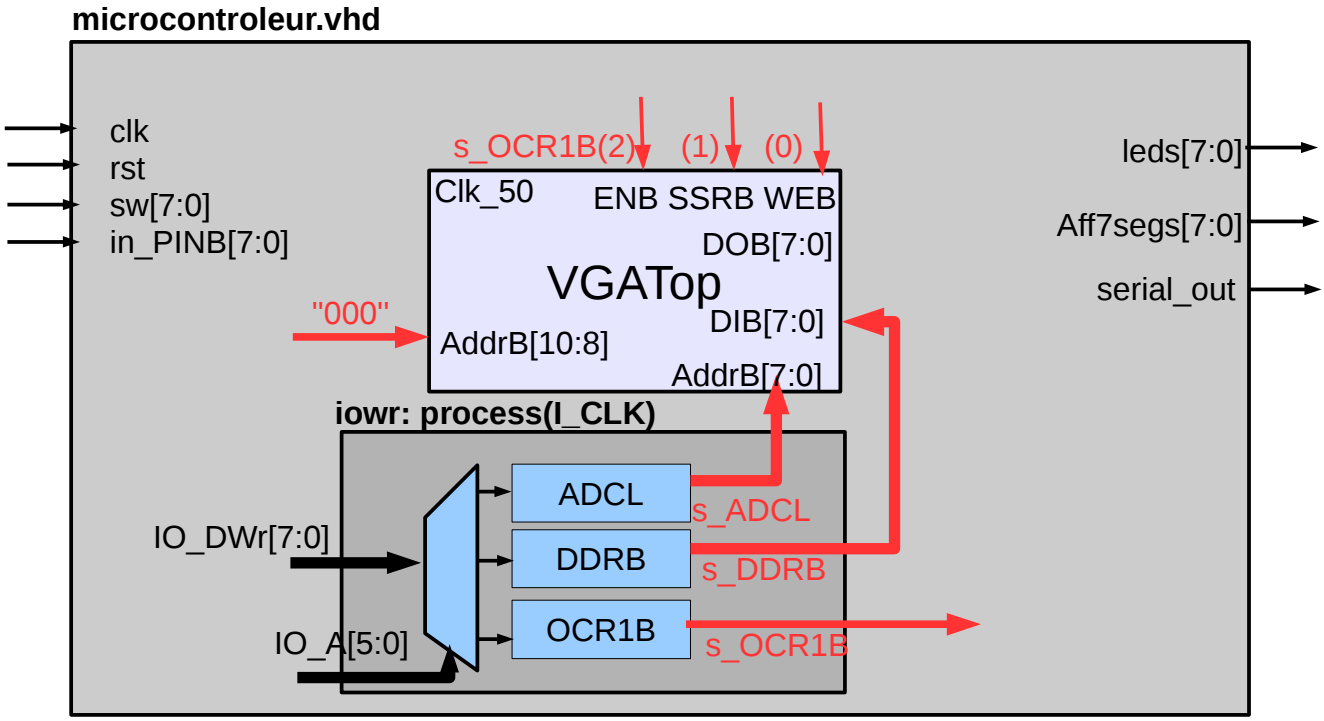

Figure 4

Pour éviter que chacun fasse son propre câblage, on vous demande absolument de relier ENB au poids (b_2) de s_OCR1B et WEB au bit b_0 (poids faible) de s_OCR1B conformément au dessin. La logique veut que SSRB soit relié au bit b_1.

Seul ce qui est en rouge est à modifier pour avoir un interfaçage du processeur **ATTiny861** à l'écran VGA. Si vous ne respectez pas le nom des fils (signaux à déclarer), assumez-le.

Faire fonctionner exofinal.c qui est un compteur décompteur (choisi avec sw0). Ajustez les adresses dans **ADCL** pour que l'affichage soit en lieu et place de vos deux "0".

A partir de maintenant plus aucun fichier VHDL ne sera développé : votre partie matérielle ne changera plus.

Travail à réaliser (4 pts exo5)

3-3) Ajouter une sortie sur la liaison série de votre compteur (seulement quand ils ont changés). On partira de exofinal.c

C'est le simple compteur de l'exercice 3-2 qui est demandé ici, avec une sortie série supplémentaire.

IV) Le compteur de passages dans l'ATTiny861

Il est temps de nous intéresser au but final du projet : le compteur de passage. Pour cela on garde l'architecture matérielle de la section précédente : plus de VHDL à écrire.

Travail à réaliser (4 pts exo6)

4-1) Reprendre le programme de l'exercice 2 et le compléter pour avoir un affichage correct sur l'écran VGA et le fonctionnement du compteur de passages.

Écrire un sous-programme d'affichage sur écran VGA de prototype :
void afficheVGA(unsigned char cnt) ;