MI41 – Final A06

Durée 2h
Documents autorisés : transparents de cours et liste des instructions ASM ARM
Les réponses non justifiées seront considérées non valables et le code non expliqué ne sera pas lu

1. Description VHDL

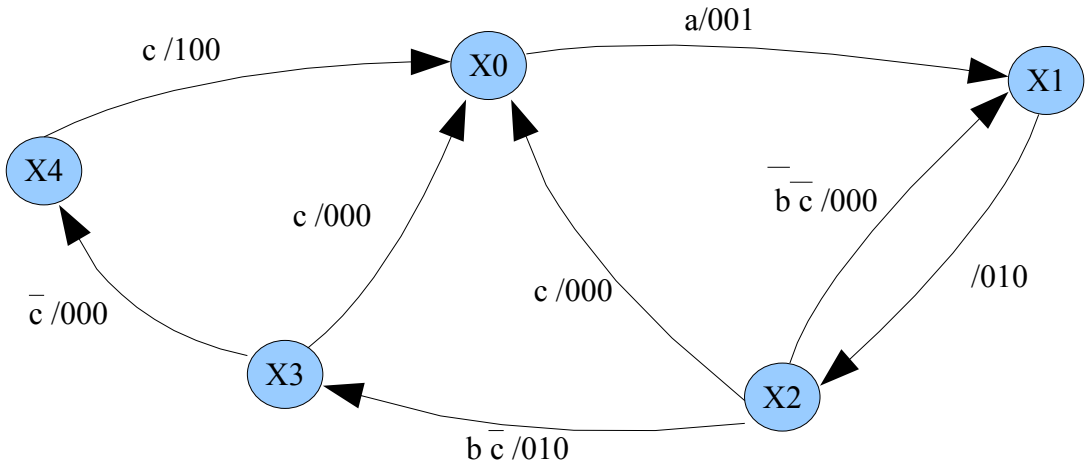
On souhaite réaliser un circuit numérique permettant de gérer un appuis sur un bouton. La description du système s'effectuera en VHDL et devra pouvoir être synthétisée.
Description du système :
Le bouton correspond à une entrée norté a, la sortie sera notée s. clk est l'horloge système.

L'appui sur un bouton entraine la mise à 1 de l'entrée a. Lorsque le bouton est appuyé, le système doit générer une impulsion positive de la durée d'une période d'horloge, lorsque le bouton est relaché, le système doit également générer une impulsion positive de la durée d'une période d'horloge, le reste du temps la sortie s est au repos (0 logique). Il devra y avoir un retour au repos entre les deux impulsions (cas d'un appui cour devant la durée d'une période d'horloge). La sortie s devra être synchronisée avec l'horloge.

Donnez la description VHDL permettant de synthétiser ce système

2. Implémentation d'un séquenceur de commande

On souhaite générer une séquence de commande sur 3 bits en fonction de trois entrées a, b et c. La séquence de commande est décrite par le graphe à état suivant :



En expliquant bien votre démarche, donnez la réalisation du circuit logique implémentant cette séquence de commandes. On utilisera des bascules D et une machine à états décodée (1 bascule active par place active)

3. ASM

Donnez le code assembleur des fonctions données ci-après. Les prototypes sont donnés en C pour faciliterla compréhension.

1. Fonction intervertissant deux données d'un tableau les adresses des deux données sont fournies en paramètre via les registre r0 (val1) et r1 (val2)

Prototype C : *void swap (int* val1, int* val2)*

2. Fonction évaluant si les bits d'une donnée mémoire, spécifiés par un masque sont tous à zéro ou non

data : adresse de la donnée fournie via r0
masque : mot de 32 bits signalant quels sont les bits à tester (coorespond au bit à 1 du masque) fourni via r1
la fonction renvoie 0 si au moins un des bits spécifiés par « masque » n'est pas à zéro autre valeur sinon.
Prototype C : *int test (int* data, int masque)*

3. Fonction donnatn le nombre d'occurence d'un entier dans un tableau

adresse tableau fourni par r0
entier dont on cherche le nombre d'occurence fourni par r1
taille du tableau (en nombre de cases) fourni par r2
retour (via r0) : nombre d'occurence
Prototype C : *int occurrence(int * tab, int valeur, int taille_tab)*