Université Bordeaux 1, Sciences et technologies
UFR Informatique et Mathématiques

Licence d'informatique
Session de septembre 2001

Architecture et Microprocesseurs (AMP)
Le 11 septembre 2001

Durée : trois heures

Tous documents autorisés sauf photocopies illégales de livres. Votre voisin n'est pas un document.

Rappel : Soyez bref et clair ! Personne n'est impressionné par la quantité !

Bon courage !

Exercice 1. (7p)

Un fabricant de *boîtes à rythmes* (une boîtier électronique capable de produire des sons qui ressemblent à ceux d'un batteur) souhaite utiliser un circuit séquentiel pour déclencher des sons divers. Pour cela, il souhaite que le circuit donne le rythme. En général, il y aura plusieurs dizaines de rythmes possibles, mais pour simplifier l'exercice, nous allons supposer qu'il n'y en a que deux. Le choix entre les deux est fait par une entrée externe (un interrupteur par exemple). La sortie de la boîte est constituée de deux bits, interprétés comme un nombre binaire entre 0 et 3 indiquant la position dans une ou plusieurs mesures de musique.

Si l'interrupteur est en position 0, la boîte donne un rythme déterminé par une seule mesure de 4 temps indiqué par la suite de sorties 0, 1, 2, 3, 0, 1, 2, 3,. . ..

Si par contre l'interrupteur est en position 1, la boîte donne un rythme déterminé par *deux mesures*, la première de 3 temps et la deuxième de 2 temps, comme ceci : 0, 1, 2, 0, 1, 0, 1, 2, 0, 1, ...

(a) **[4p]** Donner la table d'états du circuit (indication : vous avez besoin d'une sortie invisible pour déterminer si la boîte est dans la première ou dans la deuxième mesure quand l'interrupteur est en position 1).

(b) **[3p]** Réaliser le circuit.

Exercice 2. (13p)

L'instruction `case` ou `switch` d'un langage de programmation impératif comme Pascal ou C est traditionnellement implémentée par une *table de sauts*. Il s'agit d'un tableau contenant des adresses de morceaux de code. L'indice i du tableau contient l'adresse du début du morceau de code à exécuter si la valeur de l'expression de test

est l'entier i.

Pour que l'architecture soit capable d'exécuter le code généré par le compilateur pour l'instruction `case` ou `switch` assez rapidement, c'est une bonne idée de la munir d'une instruction JUMPIX prenant un seul argument, l'adresse du premier élément du tableau. Appelons cette adresse a. L'effet de l'instruction JUMPIX est de mettre dans le compteur ordinal (PC) le contenu de la mémoire à l'adresse $a+i$, où i est la valeur contenue dans le registre R1, autrement dit : $PC = M[a+i]$.

(a) [**4p**] Rajouter les circuit nécessaires pour la réalisation de l'instruction JUMPIX à l'architecture donnée en annexe.

(b) [**3p**] Réaliser le microprogramme de l'instruction JUMPIX.

(c) [**6p**] Donner le code généré par le compilateur pour l'architecture donnée en annexe pour l'instruction :

```
switch(x + y)
  {
    case 0: x = 0; break;
    case 1: y = 0; break;
    case 2: x = 0; y = 0; break;
  }
```

On suppose que les valeurs des deux variables `x` et `y` se trouvent sur des adresses fixes en mémoire, et que le compilateur sait que la valeur de l'expression `x + y` est comprise entre 0 et 2.

Vous pouvez vous servir de l'ensemble des instructions présentées en cours, ainsi que de la nouvelle instruction JUMPIX.