Université de Rennes 1 mardi 14 mars 2006

IFSIC

PREMIÈRE ANNÉE DU DIIC

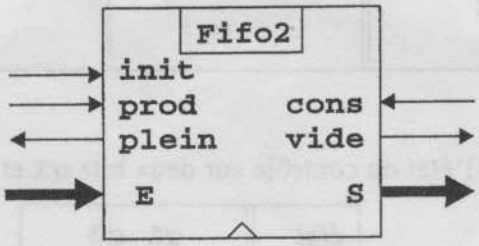
Examen 2 du module ARC1

Durée de l'épreuve : 2 heures
Le sujet comporte 4 pages
Polycopié et notes de cours, TD et TP autorisés

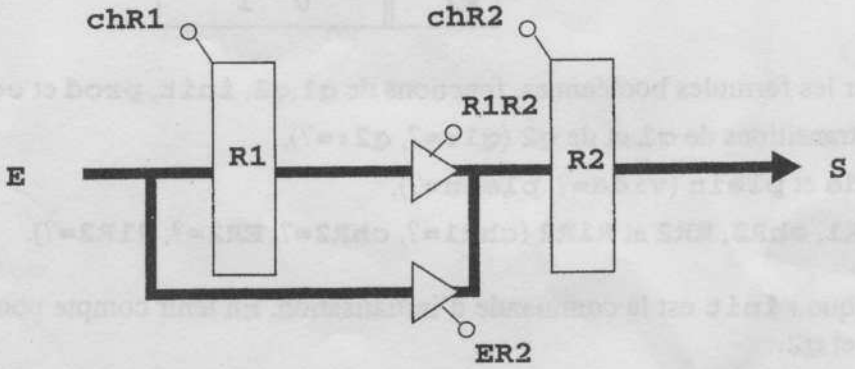
Exercice 1 (temps conseillé : 50 mn)

On veut réaliser une file d'attente à 2 places (**Fifo2**). C'est un composant capable de mémoriser jusqu'à 2 données (de 8 bits par exemple) et de les restituer dans l'ordre où elles ont été introduites :

- l'entrée **E** permet de présenter une donnée à introduire,
- la sortie de **S** présente la plus ancienne donnée présente (s'il y en a une),
- la sortie **vide** indique si la file est vide,
- la sortie **plein** indique que la file est pleine (contient 2 éléments),
- les entrées **init**, **prod** et **cons** sont trois commandes mutuellement exclusives :
  - **init** initialise la file à vide,
  - **prod** (pour "produire") enregistre la donnée **E** dans la file (si la file n'est pas pleine).
  - **cons** (pour "consommer") retire la plus ancienne donnée présente (si la file n'est pas vide).



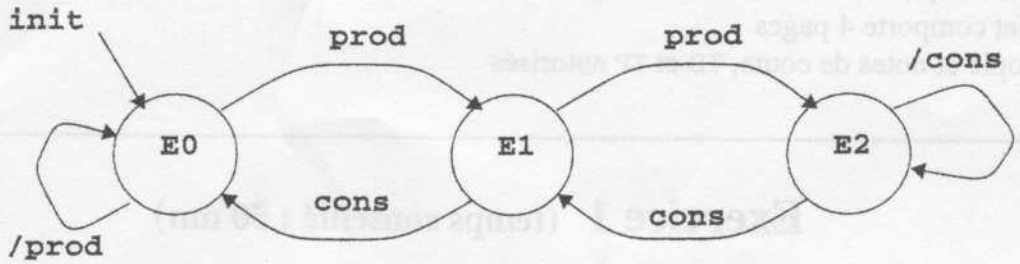
Pour réaliser cette file, on la décompose en une partie "traitement" et une partie "contrôle". Voici le schéma de la partie traitement :



Deux registres **R1** et **R2**, dotés de commandes de chargement, permettent de mémoriser les données. Le registre **R2** contient la plus ancienne donnée présente (quand il y en a une).

CHAP 10

La partie contrôle est réalisée par un automate à trois états **E0**, **E1** et **E2**. Les états ont la signification suivante :
E0 : la file est vide (ne contient aucune donnée),
E1 : la file contient une donnée,
E2 : la file est pleine (contient deux données).



Question 1.1
Les commandes **chR1**, **chR2**, **ER2** et **R1R2** dépendent de l'état et des commandes **prod** et **cons**. Indiquer, en remplissant le tableau suivant, quelle commandes doivent être générées selon l'état et les commandes **prod** et **cons**.

état	**prod**	**cons**
E0		
E1		
E2		

Question 1.2
On décide de coder ainsi l'état du contrôle sur deux bits **q1** et **q2** :

état	**q1**	**q2**
E0	1	0
E1	0	0
E2	0	1

Donner les formules booléennes, fonctions de **q1**,**q2**, **init**, **prod** et **cons**, pour :

- les transitions de **q1** et de **q2** (**q1:=**?, **q2:=**?),
- **vide** et **plein** (**vide=**?, **plein=**?),
- **chR1**, **chR2**, **ER2** et **R1R2** (**chR1=**?, **chR2=**?, **ER2=**?, **R1R2=**?).

Remarque : **init** est la commande d'initialisation. En tenir compte pour rédiger les formules de **q1** et **q2**.

chap 10

Exercice 2 (temps conseillé : 70 mn)

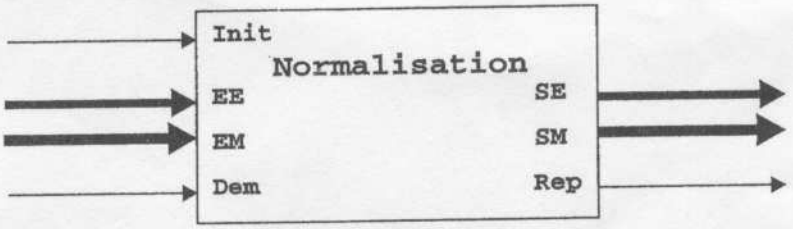
On représente des nombres réels en virgule flottante sur 32 bits comme suit (ce n'est pas la représentation standard IEEE, elle est volontairement plus simple) :

- un champ **E** de 8 bits, l'exposant, représente *en complément à 2* un entier relatif e,
- un champ **M** de 24 bits, la mantisse, représente ***en base 2*** un nombre entier positif ou nul m.
- l'interprétation de **<E,M>** est le nombre réel : $(m/2^{23}) \times 2^e$

Exemple : **X = 000000101 10001100000000000000000**
représente le nombre réel $x=(1/2 + 1/32 + 1/64)\times 2^5 = 16+1+0{,}5 = 17{,}5$

Une représentation de nombre réel est dite ***normalisée*** si le chiffre de poids fort de la mantisse (le bit 23 de **M**) est ***non nul***.
Exemple : **X' = 000000111 00100011000000000000000**
X' n'est pas normalisé.
La ***normalisation*** consiste à transformer la représentation de manière à obtenir une représentation normalisée du même nombre. Ainsi, pour normaliser X', il faut décaler sa mantisse de 2 positions vers les poids forts, ce qui revient à la multiplier par 4, et on corrige en retranchant 2 de son exposant, ce qui donne la version normalisée :
X = *normalisation*(X') = 000000101 10001100000000000000000

On veut réaliser un circuit **Normalisation** qui réalise la normalisation de la représentation de nombres réels. Ceci peut être réalisé par l'algorithme ci-après qui consiste à décaler la mantisse tout en décrémentant l'exposant jusqu'à ce que le bit 23 de **M** soit égal à 1.
Le brochage du circuit **Normalisation** est illustré sur la figure suivant :



La machine s'utilise selon un protocole classique de demande-réponse entrelacées : l'utilisateur lance une normalisation en affichant les données sur les entrées **EE** (exposant) et **EM** (mantisse) accompagné de la mise à 1 du signal **Dem**. Lorsque le résultat est calculé, la machine l'affiche sur les sorties **SE** (exposant) et **SM** (mantisse) accompagné de la mise à 1 du signal **Rep**.

```
jqa toujours faire

  jqa demandeNormalisation faire
       Rep:=0, E:=EE, M:=EM, quand Dem=1 sortir
  fait;

  jque TentativeNormalisationTerminée faire
      quand M23=1 ou E=-128 ou M=0 sortir;
     E:=E-1, M:=2*M
  fait;

  jqa résultatConsulté faire
      Rep:=1, SE=E, SM=M, quand Dem=0 sortir
  fait

fait
```


CHAP 10

Remarque : dans certains cas, la normalisation n'est pas possible, dans le cas où m=0 ou bien dans le cas où l'exposant nécessaire est "trop négatif" pour être représentable sur 8 bits. La machine délivre alors un résultat "au mieux". L'utilisateur saura que le résultat n'a pas pu être normalisé car dans ce cas le bit 23 de **SM** sera nul.

Question 2.1
Donner le schéma de l'unité de traitement du circuit **Normalisation**.
(On dispose pour **M** d'un registre à décalage doté d'une commande de chargement **chM** et d'une commande de décalage **decM**).

Question 2.2
Donner le diagramme d'états de l'unité de contrôle du circuit **Normalisation**.

Question 2.3
Donner le schéma de l'unité de contrôle réalisée à l'aide d'un codage de l'état "machine à jeton".