Partie contrôle

1. **Micro-instructions**

   On considère l'architecture de la figure 1.

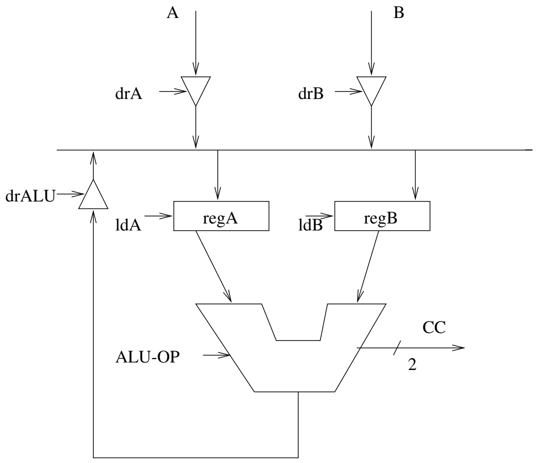

Figure 1 : chemin de données pour le pgcd

Les signaux de contrôle de ce chemin de données sont :

- le signal de contrôle de l'ALU "alu-op" : si alu-op = 0, alors le résultat est regA-regB, sinon regB-regA.
- les signaux de chargement des registres lda (load du registre regA), ldb (load du registre regB),
- les signaux de contrôle des portes 3-états dralu, dra, drb : si dralu=1 (resp dra, drb) alors la sortie de l'ALU (resp. regA, regB) passe sur le bus.

Quel est l'effet de la micro-instruction alu-op=0, ldA=1, ldB=0, dralu=0, dra=1, drb=0 ?

Quel est l'effet de la micro-instruction alu-op=1, ldA=1, ldB=1, dralu=1, dra=0, drb=0 ?

2. **Calcul du PGCD**

   On veut calculer sur le chemin de donnnées de la figure 1 le PGCD de a et b par l'algorithme d'Euclide:

```
while a != b do
  if a > b then a <- a - b
           else b <- b - a
return(a)
```

Donner le diagramme de transition d'état d'une FSM qui commande le chemin de données de la figure 1 pour réaliser l'algorithme souhaité.

Les sorties de la FSM sont alu-op, dralu, dra, drb, lda, ldb et ready qui indique que le calcul est terminé.

Les entrées de la FSM sont le flag CC qui indique si le résultat est positif (CC=01), négatif (CC=10) ou nul (CC=00), ainsi qu'un signal start qui indique au chemin de données de commencer le calcul.

Le fonctionnement de la FSM est le suivant :

- Tant que start=0, il ne se passe rien,
- Quand start=1, le calcul du pgcd commence et n'est pas interrompu, quelle que soit la valeur de start,
- Quand le calcul est en cours, le signal ready=0,
- Quand le résultat est disponible, le signal ready=1.

3. **Division entière**

Ecrire le diagramme de transition d'état d'une partie contrôle qui effectue la division entière de A par B sur le chemin de données de la figure 2 (R est le reste, Q est le quotient). La sortie N du soustracteur est à 1 si le résultat du calcul est négatif. La partie contrôle émet un signal "ready" qui est nul en cours de calcul et égal à 1 quand les données sont disponibles sur R et Q. Elle prend en entrée un signal start qui indique de commencer le calcul.

Les signaux de commande du chemin de données sont: lA, lB, lS, lQ qui chargent respectivement les registres RegA, RegB, RegS et RegQ, m qui commande le multiplexeur mux (quand m=1 A passe en entrée de RegA), resetQ qui initialise le registre RegQ à 0.

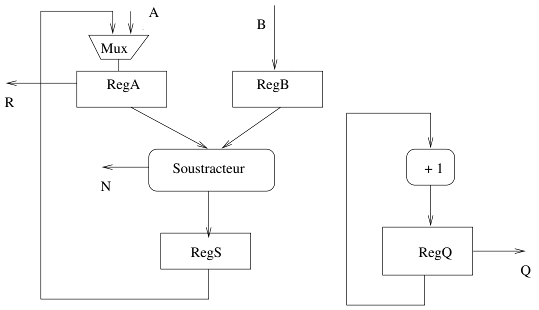

Figure 2 : chemin de données pour la division entière

4. **Partie contrôle de PROCSI**

On considère les micro-instructions de PROCSI de la figure 3.

- Quels sont les effets des micro-instructions m4, m5, m6, m7 et m8 (voir sur le document "description de PROCSI" la définition des signaux).
- Quel est l'effet de la succession des micro-instructions m0 m1 m2 m3 ?
- On souhaite réaliser la partie contrôle gérant le séquencement des micro-instructions pour le sous ensemble suivant du jeu d'instructions de PROCSI:

  instructions "add", "sub", "and" en mode "registre/registre", "immédiat/registre" et "direct/registre" instructions “load” et “store” en mode “direct/registre”.

Donner le diagramme de transition d’état qui permet d’exécuter les instructions précédentes. Les entrées sont ir et sr, les états sont les numéros de micro-instruction.

no	lrd1	lrd2	lir	lrop	lrad	incpc	mux	f1f2f3m	r/wreg	csreg	r/w	as	ds	adbus
0					1						1	0		000
1						1					1	0	0	
2											1	0	0	
3			1								1	0	0	010
4	1						1		1	0				011
5		1					0		1	0				011
6							1	1101	0	0				001
7							1	1111	0	0				001
8							1	1000	0	0				001
9					1						1	0		000
10						1					1	0	0	
11											1	0	0	
12				1							1	0	0	010
13		1												100
14					1						1	0		100
15											1	0	0	
16											1	0	0	
17		1									1	0	0	010
18							1	0110	0	0				001
19					1						0	0		100
20								0110			0	0	0	001
21								0110			0	0	0	001
22								0110			0	0	0	001

Figure 3 : micro-instructions de PROCSI

On pourra utiliser les résultats suivants (que vous pouvez vérifier grâce au tableau précédent) :

```
Effet de la suite m9, m10, m11, m12 :
    rad <-  pc
    rop <- mem[rad]
    pc <- pc + 1
Effet de m13 :
    rd2 <- rop
Effet de la suite m14, m15, m16, m17 :
    rad <- rop
    rd2 <- mem[rad]
Effet de m18 :
    reg[ir(0..2)] <- rd2
Effet de la suite m19, m20, m21, m22:
    rad <- rop
    mem[rad] <- rd2
```

- Pour quelles instructions la partie contrôle de PROCSI utilise-t-elle l’entrée SR ?