Année 2003 – 2004   Session de Janvier 2004
LST Informatique

Épreuve d'Architecture de l'ordinateur

Date : le 24 janvier 2004   Durée : 1h30

Tous documents légaux autorisés.

Votre voisin n'est pas un document

Vous devez répondre directement sur le sujet

Épreuve de M Strandh

Répondre directement sur le sujet dans les cadres prévus à cet effet. Aucune réponse ou partie de réponse en dehors des cadres ne sera prise en compte.

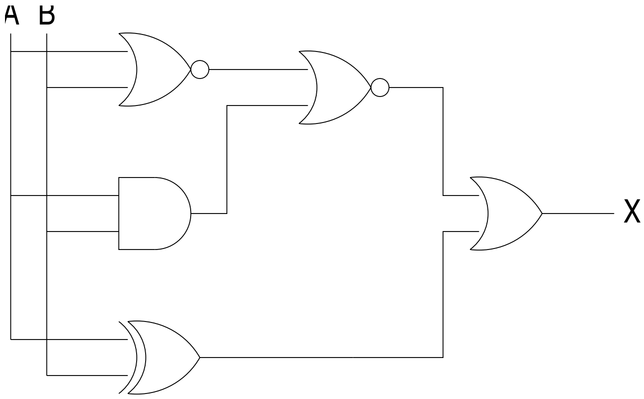
Écrire votre numéro d'anonymat sur chaque feuille du sujet.

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

Bon courage !

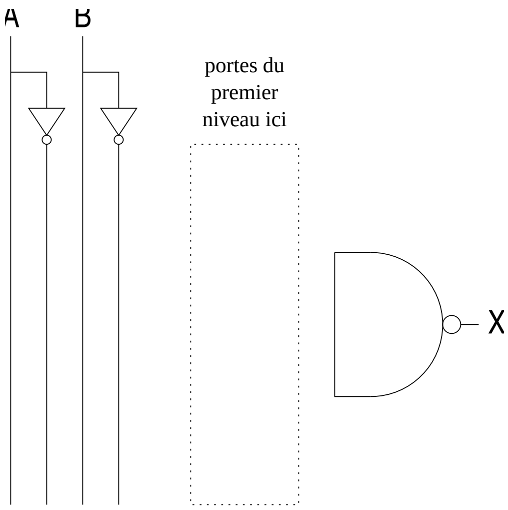
Exercice 1 (2 p)

Voici un circuit combinatoire avec deux entrées (`A` et `B`) et une sortie (`X`) :



Réaliser un circuit à deux niveaux utilisant uniquement des portes **non-et** et des portes **non** équivalent au circuit ci-dessus.

Réponse :



Exercice 2 (2 p)

Expliquer en une seule phrase ce que fait le circuit séquentiel correspondant à la table d'états suivante :

x_3	x_2	x_1	x_0	a	x'_3	x'_2	x'_1	x'_0
0	0	0	0	0	0	0	0	1
0	0	0	0	1	0	0	1	1
0	0	0	1	0	0	0	1	0
0	0	0	1	1	0	0	1	1
0	0	1	0	0	0	1	0	0
0	0	1	0	1	0	0	1	1
0	0	1	1	0	0	1	0	0
0	0	1	1	1	0	1	1	0
0	1	0	0	0	0	1	0	1
0	1	0	0	1	0	1	1	0
0	1	0	1	0	0	1	1	1
0	1	0	1	1	0	1	1	0
0	1	1	0	0	0	1	1	1
0	1	1	0	1	1	0	0	1
0	1	1	1	0	1	0	0	0
0	1	1	1	1	1	0	0	1
1	0	0	0	0	1	0	1	0
1	0	0	0	1	1	0	0	1
1	0	0	1	0	1	0	1	0
1	0	0	1	1	1	1	0	0
1	0	1	0	0	1	0	1	1
1	0	1	0	1	1	1	0	0
1	0	1	1	0	1	1	0	1
1	0	1	1	1	1	1	0	0
1	1	0	0	0	1	1	0	1
1	1	0	0	1	1	1	1	1
1	1	0	1	0	1	1	1	0
1	1	0	1	1	1	1	1	1
1	1	1	0	0	0	0	0	1
1	1	1	0	1	1	1	1	1
1	1	1	1	0	0	0	0	1
1	1	1	1	1	0	0	0	0

Réponse :

Exercice 3 (4 p)

Annexe 1 montre une modification de l'architecture traitée en cours. Cette modification permet de charger un nouveau registre à partir de la mémoire, de décrémenter celui-ci et de vérifier si son contenu est égal à zéro. Annexe 2 montre le nouveau contenu du circuit pour la gestion des conditions.

Voici le micro programme d'une instruction (appelée `JZD`) qui profite de cette modification :

Cycle	Micro opérations dont la valeur est égale à 1
1	5 3 9 7
2	1 10 21 23

Donner une description de l'instruction destinée aux programmeur de l'architecture. [2p]

Réponse :

En supposant l'existence d'une autre instruction (`LDD`) utilisant la micro opération 22 pour charger une valeur immédiate (comme `LDIMM`) dans le nouveau registre, montrer le code généré par un compilateur pour une boucle dont le nombre d'itérations est connu à la compilation. [2p]

Réponse :

Exercice 4 (6 p)

Voici un sous programme en code *assembleur* (les *noms* plutôt que les codes des instruction sont employés, et les adresses sont *symboliques* plutôt que numériques) :

```
mystere: lds 2
         copy
         jz out
         ldimm -1
         add
         push
         lds 2
         copy
         push
         jsr mystere
         addsp 2
         push
         lds 2
         copy
         push
         jsr mult
         addsp 2
         ret
out:     ldimm 1
         copy
         ret
```

On suppose l'existence d'un sous programme à l'adresse (symbolique) `mult` qui, étant donné deux arguments dans la pile, calcule leur produit dans R1.

En supposant que la première instruction du sous programme `mystere` se trouve à l'adresse 40 (en base 10) et celle du sous programme `mult` à l'adresse 150 (en base 10), quel est le contenu de la mémoire (sous la forme de suite d'octets) entre l'adresse 40 et l'adresse 70 (se servir de l'annexe 3 pour les codes des instructions) ? [3p]

(réponse sur la page suivante)

Réponse :

Adresse	Contenu (en base 10)
40	
41	
42	
43	
44	
45	
46	
47	
48	
49	
50	
51	
52	
53	
54	
55	
56	
57	
58	
59	
60	
61	
62	
63	
64	
65	
66	
67	
68	
69	
70	

Combien de paramètres sont attendus par le programme `mystere` ? [1p]

Réponse :

Quelle est la fonction calculée par le sous programme `mystere` ? [2p]

Réponse :

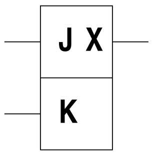
Exercice 5 (6 p)

Dans la littérature, on parle souvent de plusieurs différents types de bistables.

Ainsi un *bistable* `JK` est un circuit séquentiel avec deux entrées (appelées `J` et `K`) et une sortie `X` dont voici la table d'états :

J	K	X	X'
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

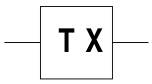
Voici le symbole utilisé pour le bistable `JK` (l'horloge n'est pas indiquée) :



Un *bistable* `T` est un circuit séquentiel avec une entrée (appelée `T`) et une sortie `X` dont voici la table d'états :

T	X	X'
0	0	0
0	1	1
1	0	1
1	1	0

Voici le symbole utilisé pour le bistable `T` (l'horloge n'est pas indiquée) :



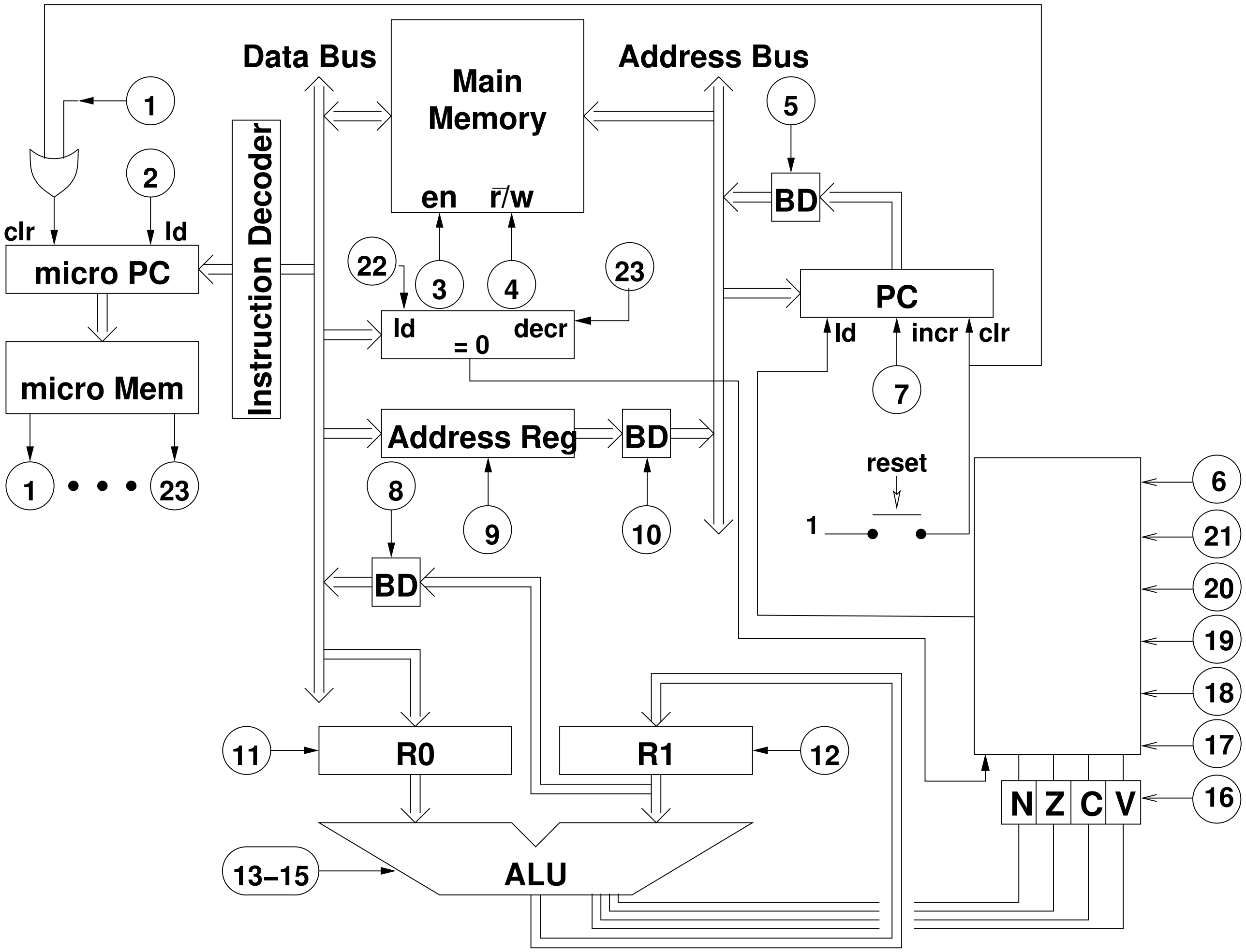
(l'exercice continu sur la page suivante)

Montrer par un dessin comment construire un bistable `JK` à partir d'un bistable `T` et un circuit combinatoire à deux niveaux. [3p]

Réponse :

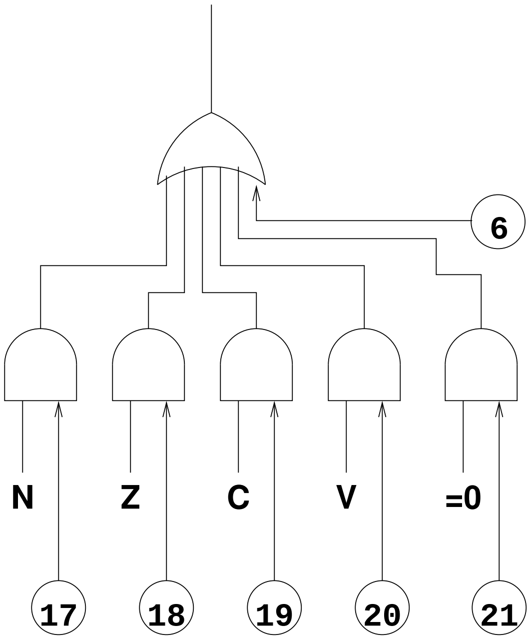
Montrer par un dessin comment construire un bistable `D` à partir d'un bistable `T` et une seule porte logique supplémentaire. [3p]

Réponse :



Data Bus
Main Memory
Address Bus
en
r̄/w
Instruction Decoder
micro PC
micro Mem
clr
ld
1
2
3
4
5
22
23
ld
decr
= 0
BD
PC
ld
incr
clr
7
reset
1
Address Reg
BD
8
9
10
BD
6
21
20
19
18
17
16
11
R0
R1
12
N
Z
C
V
13–15
ALU
1 • • • 23

Annexe 2



Annexe 3

Nom	Code	Description
nop	0	ne rien faire
ldimm	1	chargement immédiat
ld	2	chargement
st	3	stockage
copy	4	copier R0 dans R1
shl	5	décalage de R1 à gauche
shr	6	décalage de R1 à droite
add	7	R1 reçoit la somme de R0 et R1
sub	8	R1 reçoit la différence de R1 et R0
and	9	R1 reçoit le *et* logique entre R0 et R1
or	10	R1 reçoit le *ou* logique entre R0 et R1
not	11	R1 reçoit la négation de R1
jal	12	saut non conditionnel
jn	13	saut si le résultat négatif
jz	14	saut si le résultat zéro
jv	15	saut si débordement
jc	16	saut si retenue
jsr	17	appel à sous programme
ret	18	retour de sous programme
push	19	empiler R1
pop	20	dépiler dans R0
lds	21	chargement d'une valeur dans la pile
sts	22	stockage dans la pile
addsp	23	additionner une constante à SP