Architecture des ordinateurs et systèmes d'exploitation

Corrigé du TD 4: Une architecture simple

Marcel Bosc Christophe Dehlinger Arnaud Giersch Mathieu Haefele
Benoît Meister Nicolas Passat

On considère l'architecture 32 bits simplifiée composée des éléments suivants :
- Une UAL (Unité Arithmétique et Logique).
- Les registres :
  - R_1, R_2, …, R_{32} (registres de données) ;
  - R_{op1} et R_{op2} (registres opérandes de l'UAL) ;
  - R_{res} (registre résultat de l'UAL) ;
  - Flags (registre des flags, composé de 4 bits : Z, C, O, S) ;
  - IP (registre adresse de l'instruction courante) ;
  - RI (registre contenant l'instruction courante) ;
  - RTA (registre tampon adresse, contenant l'adresse d'un mot à lire/écrire en mémoire) ;
  - RTD (registre tampon données, contenant un mot à lire/écrire en mémoire).
- Deux types de signaux :
  - R/W (signal déclenchant la lecture/écriture en mémoire) ;
  - E (signal activant l'UAL).
- Une mémoire pour les données et les instructions (mots de 4 octets).

Un jeu d'instructions réduit est associé à cette architecture :

Instruction	Syntaxe	Signification
Addition	add R_x, R_y, R_z	$R_x \leftarrow R_y + R_z$
Soustraction	sub R_x, R_y, R_z	$R_x \leftarrow R_y - R_z$
Et	and R_x, R_y, R_z	$R_x \leftarrow R_y$ and R_z
Ou	or R_x, R_y, R_z	$R_x \leftarrow R_y$ or R_z
Chargement	load R_x, R ou Adresse	$R_x \leftarrow \text{Mem}(\text{R ou Adresse})$
Enregistrement	store R_x, R ou Adresse	$\text{Mem}(\text{R ou Adresse}) \leftarrow R_x$
Comparaison	cmp R_x, R_y	$Z \leftarrow 0$ si $R_x = R_y$, 1 sinon
Branchement conditionnel	bz R ou Adresse	Branchement si $Z = 0$
Branchement	ba R ou Adresse	Branchement dans tous les cas

Remarques :
- Les instructions sont toutes sur 4 octets ;
- R_α correspond à un des registres R_1, …, R_{32} ;
- Adresse correspond à une adresse sur 16 bits. Comme les adresses doivent être sur 32 bits, les 16 bits de poids fort sont alors mis à 0.
- Comme les mots mémoire sont codés sur 4 octets, toutes les adresses de mots doivent être des multiples de 4.

1. Donnez la significations des flags Z, C, O, S.

   Correction :
   - *Z : Flag égal à 0 si le résultat de l'opération de l'UAL est nul ;*
   - *C : Flag de retenue (opération arithmétique) ;*
   - *O : Flag d'overflow (opération arithmétique) ;*
   - *S : Flag de signe (opération arithmétique) ;*

2. Quelle est l'adresse la plus grande qui peut être utilisée ?

   Correction : *Les données et adresses sont codées sur 32 bits, l'adresse la plus grande est donc* $2^{32}-1$.

3. Quelle est la taille mémoire utilisable (i.e. adressable), en bits ?

   Correction : *Un octet étant composé de* $8 = 2^3$ *bits, le nombre de bits utilisable est de* $2^{32} \times 2^3 = 2^{35}$.

4. Quel est le plus grand entier non signé (resp. signé) qui peut être manipulé ?

   Correction : *Le plus grande entier non signé (resp. signé) codable sur 32 bits est* $2^{32}-1$ *(resp.* $2^{31}-1$*).*

5. Pour chaque type d'instruction, donnez les registres concernés.

   Correction : *Les registres IP, RI, RTA et RTD sont toujours concernés puisqu'ils permettent de charger l'instruction suivante. Le tableau suivant ne tient donc pas compte de cette fonctionnalité.*

Instruction	$R_1, \ldots, R_{32}$	R_{op1}, R_{op2}	R_{res}	*Flags*	*IP*	*RI*	*RTA*	*RTD*
add	*1, 2 ou 3*	*X*	*X*	*X*	*X*			
sub	*1, 2 ou 3*	*X*	*X*	*X*	*X*			
and	*1, 2 ou 3*	*X*	*X*	*X*	*X*			
or	*1, 2 ou 3*	*X*	*X*	*X*	*X*			
load	*1 ou 2*				*X*		*X*	*X*
store	*1 ou 2*				*X*		*X*	*X*
cmp	*1 ou 2*	*X*		*X*	*X*			
bz	*0 ou 1*			*X*	*X*			
ba	*0 ou 1*				*X*			

6. Donnez les étapes nécessaires pour chacune des instructions suivantes :
   - load R_5, 1016
   - store R_2, R_4
   - add R_4, R_1, R_{12}
   - sub R_6, R_6, R_5
   - and R_1, R_2, R_3
   - or R_1, R_8, R_{11}
   - cmp R_4, R_6
   - bz 4096
   - ba 4000

   Correction : *N.B. : Les trois dernières opérations à effectuer sont toujours :*

   1. *copier IP dans RTA*
   2. *signal R*
   3. *copier RTD dans RI*

   - *load* R_5*, 1016*
     1. *placer 1016 dans RTA*
     2. *signal R*
     3. *copier RTD dans* R_5
     4. *copier* $IP + 4$ *dans IP*
   - *store* R_2*,* R_4
     1. *copier* R_4 *dans RTA*
     2. *copier* R_2 *dans RTD*
     3. *signal W*
     4. *copier* $IP + 4$ *dans IP*
   - *add* R_4*,* R_1*,* R_{12}

1. *copier R_1 dans R_{op1}*
2. *copier R_{12} dans R_{op2}*
3. *signal E : mise à jour de R_{res} et Flags*
4. *copier R_{res} dans R_4*
5. *copier $IP+4$ dans IP*

– *sub R_6, R_6, R_5*
1. *copier R_6 dans R_{op1}*
2. *copier R_5 dans R_{op2}*
3. *signal E : mise à jour de R_{res} et Flags*
4. *copier R_{res} dans R_6*
5. *copier $IP+4$ dans IP*

– *and R_1, R_2, R_3*
1. *copier R_2 dans R_{op1}*
2. *copier R_3 dans R_{op2}*
3. *signal E : mise à jour de R_{res} et Flags*
4. *copier R_{res} dans R_1*
5. *copier $IP+4$ dans IP*

– *or R_1, R_8, R_{11}*
1. *copier R_8 dans R_{op1}*
2. *copier R_{11} dans R_{op2}*
3. *signal E : mise à jour de R_{res} et Flags*
4. *copier R_{res} dans R_1*
5. *copier $IP+4$ dans IP*

– *cmp R_4, R_6*
1. *copier R_4 dans R_{op1}*
2. *copier R_6 dans R_{op2}*
3. *signal E (soustraction) : mise à jour de Flags*
4. *copier $IP+4$ dans IP*

– *bz 4096*
1. *si $Z=0$ alors placer 4096 dans IP sinon copier $IP+4$ dans IP*

– *ba 4000*
1. *placer 4000 dans IP*

7. On considère l'état mémoire suivant :

Adresse	Instructions	Adresse	Données
100	load R_4, 0	0	1
104	load R_1, 12	4	10
108	load R_2, 4	8	5
112	load R_3, 8	12	0
116	cmp R_1, R_2		
120	bz 148		
124	add R_1, R_1, R_4		
128	add R_5, R_2, R_5		
132	sub R_6, R_6, R_3		
136	cmp R_1, R_2		
140	bz 148		
144	ba 124		
148	store R_5, 4		
152	store R_6, 8		

En supposant que tous les registres R_1, …, R_{32} sont initialement à 0, donnez le contenu de ces registres ainsi que le contenu de la mémoire après exécution.

Correction :

- $R_1 = 10$;
- $R_2 = 10$;
- $R_3 = 5$;
- $R_4 = 1$;
- $R_5 = 100$;
- $R_6 = -50$;
- $R_7, \ldots, R_{32} = 0$.

Adresse	*Données*
0	*1*
4	*100*
8	*-50*
12	*0*