TD n° 4 : Fonctions et procédures

On utilise dans ce TD les jeux d'instructions MIPS32 et ARM.

1. Procédures simples

Soit le programme C suivant

```
int a, b, c ;
main(){
b=10; c=15;
a= max (b,c) ;
}

int max (int x, int y) {
if (x > y) return x ;
      else return y;
}
```

a) Écrire le programme appelant et la fonction en assembleur MIPS. (Les variables A, B, C sont rangées dans des cases mémoire successives).
b) Même question en assembleur ARM en passant les paramètres par registres, puis en les passant par la pile.

2. Procédures imbriquées

Soit le programme C suivant, qui trie un tableau de N octets signés en utilisant des procédures.

```
#include <stdio.h>
char v[10];

int main()
{
      v[0]=100;v[1]=80;v[2]=70;v[3]=60;v[4]=55;
      v[5]=40;v[6]=35;v[7]=30;v[8]=25;v[9]=10;

      tri (v,10);
}


change (char v[], int k, int m)
{
      int temp;
      temp=v[k];
      v[k]=v[m];
      v[m]=temp;
}
```

```
tri (char v[], int n)
{
            int i, j;
            for (i=n-1;i>0; i--){
                        for (j=i-1; j>=0   ; j--){
                              if (v[j] >v [i])   change (v,j, i);
                               }
                        }
}
```

Écrire le programme principal et les deux procédures en code MIPS32 et en code ARM en passant les paramètres par registres.

NB : cet exercice aborde la question des procédures imbriquées. Il est évident que la programmation serait plus simple en intégrant directement le code de change dans la procédure de TRI.

3. Annexes

MIPS32

Registres

Nom registre	Numéro	Utilisation	Sauvegarde sur appel
$zero	**0**	**Valeur 0**	
$at	**1**	**Réservé pour assembleur**	
$v0-$v1	**2--3**	**Résultats et évaluation expression**	**non**
$a0-$a3	**4--7**	**arguments**	**oui**
$t0-$t7	**8--15**	**valeurs temporaires**	**non**
$s0-$s7	**16--23**	**valeurs sauvegardées**	**oui**
$t8-$t9	**24-25**	**temporaires supplémentaires**	**non**
$gp	**28**	**pointeur global**	**oui**
$sp	**29**	**pointeur de pile**	**oui**
$fp	**30**	**pointeur de trame**	**oui**
$ra	**31**	**adresse de retour**	**oui**

ARM

Instructions pour évaluation des conditions et branchements

Instructions de comparaison	CMP Rs1, Rs2 TST Rs1, Rs2	Rs1-Rs2 → Rcc Rs1 and Rs2 → Rcc
Instructions arithmétiques avec suffixe S : ADDS, SUBS, etc	SUBS Rd, Rs1, Rs2	Rd ← Rs1 – Rs2 Positionne Rcc
Bcond (LT, LE, GT, GE, EQ, NE…)	Bcond, déplacement	Si cond, alors CP ← NCP +déplacement
BL	BL déplacement	R14 ← NCP CP ← NCP + déplacement
BLcond	BLcond déplacement	Si cond, alors { R14 ← NCP CP ← NCP +déplacement}

Conventions logicielles

Pour ARM

- R0 à R3 : registres de travail, arguments des procédures
- R4 à R8 : variables dans des registres (à sauvegarder par les fonctions appelées)
- R9 : variable registre, base statique
- R10 : variable registre, limite pile
- R11 : pointeur de trame
- R12 : registre de travail
- R13 : pointeur de pile
- R14 : registre de lien, registre de travail
- R15 : compteur de programme