Laborator 3

2.1 Moduri de adresare MIPS

Prin mod de adresare se înțelege metoda de a specifica un operand sau o adresă de memorie.

- **Adresarea imediată:** Operandul se află în corpul instrucțiunii
  De exemplu:
  ```
  addi $t0, $t1, 3 ;; instrucțiunea conține al doilea operand (3)
  and  $v0, 0x34
  ```

- **Adresarea directă:** Adresa operandului (adresa efectivă, AE) se află în corpul instrucțiunii fie sub forma unui cuvânt adresă la memorie, fie sub forma unui cuvânt număr registru din banca de registre (adresă de registru).
  ```
  move $a0, $t2     ;; instrucțiunea conține adresa operanzilor sub forma unor numere de
                    ;; de regiștrii
  ```
  In cazul arhitecturii MIPS, modul de adresare directă în care operanzii sunt valorile conținute în regiștrii se numeste **Register Addressing**.

- **Adresarea indirectă:** În corpul instrucțiunii se specifică o adresă intermediară, AI; apoi, la punctul indicat de adresa intermediară se găsește înscrisă adresa efectivă AE, se extrage adresa efectivă (ca pointer) și se citește, de la locul indicat de AE, operandul necesar în instrucțiune.
  Există două variante de adresare indirectă: prin memorie și prin registru
  ```
  jr $31            ;; adresa de memorie la care se produce saltul este conținută în
                    ;; registrul $31
  ```

  ```
  lw $t1,4($t0)     ;; valoarea încărcată in $t1 se afla la adresa ($t0+4)
  lw $t1,8($t0)
  lw $t1,16($t0)
  Inainte de a incercat instructiunile, incarcati in registrul $t0 o
  adresa din zona de stiva (Stack address) folosind instructiunile lui si
  ori.
  ```
  Pentru MIPS acest mod de adresare se numește **Base addressing** și se referă la situația în care adresa de memorie folosită pentru a aduce operandul se calculează prin însumarea unei valori conținute într-un registru și un ofset specificat în instrucțiune ca valoare imediată.

- **Adresare relativă:** Adresa efectivă, AE, se obține prin sumarea la adresa de referință a instrucțiunii (uzual adresa conținută în PC) a unei valori specificată ca un imediat.
  Acest mod de adresare pentru MIPS se numește **PC-relative addressing**
  ```
  beq $t0,$t1,16    ;; dacă regiștrii $t0 și $t1 au valori egale, noua adresă de  program se va
                    ;; calcula PC=(PC+4)+16. PC+4 reprezintă adresa următoarei  instrucțiuni
  ```

Organizarea memoriei la microprocesorul MIPS, in simulatorul SPIM:

Memory Organization

0x00400000	Code
0x10000000 - 0x10040000	Data
0x7fffeffc	Stack
0x80000180	Kernel Code
0x90000000	Kernel Data

Functii de sistem, ce pot fi utilizate in simulatorul SPIM:

System calls

Service	System Call Code	Arguments	Result
print_int	1	$a0 = integer	
print_float	2	$f12 = float	
print_double	3	$f12 = double	
print_string	4	$a0 = string	
read_int	5		integer (in $v0)
read_float	6		float (in $f0)
read_double	7		double (in $f0)
read_string	8	$a0 = buffer, $a1 = length	
sbrk	9	$a0 = amount	address (in $v0)
exit	10		

Numarul funcției se pune întotdeauna în registrul $v0.

Exemplu apelare funcție exit

```
li $v0, 10
syscall
```

Afisarea valorii unui numar intreg in consola

```
# incarcam codul functiei in registrul $v0
li $v0, 1

# in registrul t1 este valoarea pe care dorim sa o afisam
lw $a0, $t1
syscall
```

Exerciții și întrebări:

6. Incarcatii in registrii $s0 si $s1 valorile din variabilele n = 65 535 si m = 65 538. Realizati suma celor doi registrii.
   Repetati exercitiul pentru valori mai mari, pe 32 de biti: 0x80000004, 0x80000003

7. Realizati inmultirea a celor doi registrii. Unde este stocat rezultatul?

8. Realizati in limbaj de asamblare MIPS urmatoarele instructiuni de nivel inalt:

```
if(a < b){
  c = b + 1;
} else {
  c = a + 1;
}
```

9. Modificati programul de la exercitiul 8 astfel încât să avem condiția (a>=b).

10. Realizati suma unui sir de numere intregi stocate in memorie. Lungimea sirului este stocata in memorie in variabila n.
    - Stocarea unui numar in memorie se face in felul urmator:

```
.data
a: .word 5 7 9 10 12 35
n: .word 6

.text
main:
        lw $s0, n
        la $s1, a
```

.data - secțiune unde datele sunt stocate în memorie

.text - secțiunea care conține instrucțiunile și logica programului