Raport Detaliat MIPS16

Laborator AC 2020

Nume : Neagu Tudor

Grupa 30222

1.Cele 4 instructiuni suplimentare alese:

a) SLT – seteaza RF[rd] pe 1 daca RF[rs]<RF[rt], altfel pe 0

Structura instructiunii este : 000\_rs\_rt\_rd\_sa\_110

-unde rs, rd, sunt pe 3 biti iar “sa” este 1 bit.

RTL : if RF[rs] < RF [rt] RF[rd] =1 else RF[rd] =0

In asamblare: slt $3,$0,$2

b)XOR – functia face sau exclusiv intre valorile din 2 registrii

Structura instructiunii este : 000\_rs\_rt\_rd\_sa\_101

-unde rs, rt, rd, sunt pe 3 biti, sa este pe 1 bit

RTL : RF[rd]=RF[rs] xor RF[rt];

In asamblare: xor $2, $3, $2

c)SUBI- instructiunea face scadere cu imediat extins cu semn .

Structura instructiunii este : 011\_rs\_rt\_immediat

-unde rs si rt 3 biti iar immediatul pe 7 biti.

RTL abstracutl instriuctiunii : RF[rt]=RF[rs] - Sign\_ext(Imm);

In asamblare: subi $1,$3,imm;

d)ANDI– instructiunea face and cu imediat extins cu 0(logical extension) .

Structura instructiunii este : 011\_rs\_rt\_immediat

-unde rs si rt 3 biti iar immediatul pe 7 biti.

RTL abstracutl instriuctiunii : RF[rt]=RF[rs] and Zero\_ext(Imm);

In asamblare: andi $1,$3,imm;

2)Descrierea semnalelor de control

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instructiune | Reg  Dst | Reg  Write | Alu  Src | Ext  Op | Alu  Ctrl | Branch | Jump | Mem  Write | Mem  toReg |
| Add | 1 | 1 | 0 | 0 | 000 | 0 | 0 | 0 | 0 |
| Sub | 1 | 1 | 0 | 0 | 000 | 0 | 0 | 0 | 0 |
| Sll | 1 | 1 | 0 | 0 | 000 | 0 | 0 | 0 | 0 |
| Srl | 1 | 1 | 0 | 0 | 000 | 0 | 0 | 0 | 0 |
| And | 1 | 1 | 0 | 0 | 000 | 0 | 0 | 0 | 0 |
| Or | 1 | 1 | 0 | 0 | 000 | 0 | 0 | 0 | 0 |
| Xor | 1 | 1 | 0 | 0 | 000 | 0 | 0 | 0 | 0 |
| Slt | 1 | 1 | 0 | 0 | 000 | 0 | 0 | 0 | 0 |
| Addi | 0 | 1 | 1 | 1 | 001 | 0 | 0 | 0 | 0 |
| Lw | 0 | 1 | 1 | 1 | 100 | 0 | 0 | 0 | 1 |
| Sw | 0 | 1 | 1 | 1 | 101 | 0 | 0 | 1 | 0 |
| Beq | 0 | 0 | 0 | 1 | 111 | 1 | 0 | 0 | 0 |
| Subi | 0 | 0 | 0 | 1 | 110 | 0 | 0 | 0 | 0 |
| Andi | 0 | 1 | 1 | 1 | 011 | 0 | 0 | 0 | 0 |
| Jump | 0 | 0 | 0 | 0 | 010 | 0 | 1 | 0 | 0 |

3)Descrierea in cuvinte a programului

Programul face suma numerelor de la 1 la un numar dat ( in cazul de fata 4)

Int a=0,b=0,c=1,d=0;

b=4;

while(b!=d){

a=a+b;

b=b-c;

}

Trasare(cod+instructiune)

0: andi $1 $1 0 -> se initializeaza rf[1] cu 0

011\_001\_001\_0000000

1: andi $2 $2 0 -> se initializeaza rf[2] cu 0

011\_010\_010\_0000000

2: andi $3 $3 1 -> se initializeaza rf[3] cu 0

011\_011\_011\_0000001

3: andi $4 $4 0 -> se initializeaza rf[4] cu 0

011\_100\_100\_0000000

4: addi $2 $2 4 -> rf[2]=4

001\_010\_010\_0000100

5: add $1 $1 $2 -> rf[1]=rf[1] + rf[2]

000\_001\_010\_001\_0\_100

6: sub $2 $2 $3 -> rf[2]=rf[2] - rf[3]

000\_010\_011\_010\_0\_001

7: beq $2 $4 1 ->daca rf[2] = rf[4] (care e 0) se sare peste jump(adica se iese din bucla)

111\_100\_010\_0000001

8: jump 5 ->se sare la instr 5

010\_000\_0000000101

Rezultatul se va memora in RF[1]

Programul a fost testat pe placa si merge bine

Schematic:

O imagine care conține captură de ecran

Descriere generată automat