**Omerovic Muhamed 18772 Logicki Dizaj Zadaca 3**

**2. Zadatak**

a) Projektovati 4-struko grupno-asocijativni keš kapaciteta 64K sa blokom veličine 16 riječi (64B) i 32- bitnom adresom.

**Cap = 64K**

**Ass = 4**

**Block =64B**

**Adr = 32**

**Propracun :**

N =

8

6

18

**Rezultat:**

**Tag = 18b , index = 8b, offset=6b**

b) Projektovati direktno preslikani keš kapaciteta 64K sa blokom veličine 16 riječi (64B) i 64-bitnom adresom.

**Adr = 64**

**Block = 64B**

**Cap = 64K**

**Ass = 1**

**Propracun:**

1K

10

6

48

**Rezultat:**

**Tag = 48b , index = 10b, offset=6b**

**Omerovic Muhamed 18772 Logicki Dizaj Zadaca 3**

**3. Zadatak**

Koristeći mikroarhitekturu oglednog procesora, napisati mikrokod za instrukciju MULCO (ac := ac \* x) koja će pomnožiti sadržaj AC registra sa konstantom specificiranom u 8 donjih bita instrukcije (x). Primjer upotrebe instrukcije: MULCO A7

**80: A := smask; //0000 0000 1111 1111**

**81: A := band(IR,A); //0000 0000 xxxx xxxx**

**82: B := AC;**

**83: ALU := A; if Z then goto 0;**

**84: ac := ac + B;**

**85: A := A + (-1); goto 83;**

**Objasnjenje:**

**Sa instrunkcijama 80. I 81.** dobivamo posljednjih 8 bita sa Instrunkcijskog Registra (IR) I smjestamo ga u registar A, to nam oznacava mnozilac sa kojim treba da pomnozimo vrijednost na akumulatoru (AC).

**Linija 82.** smjesta pocetnu vrijednost akumulatora u registar B. Posto nemamo instrukciju za mnozenje to mozemo tretirati kao sabiranje akumulatora samim sa sobom n puta to nam je ekvivalnet ac \* x.

**Linija 83.** smjesta trenutni brojac iz registra A u ALU I provjerava da li je Zero flag 0 ako jeste znaci da smo zavrsili sa instrukcijom a ako nije nastavljamo dalje sa izvravanjem.

**Linija 84.** sabira trenutnu vrijednsot akumulatora sa svojom pocetnom vrijednosti.

**Linija 85**. oduzima od vrijednosti sabirnice A , odnosno dekrementira vrijednost sabirnice A I vracamo se na liniju 83 gdje cemo opet provjeravati da li smo zavrsili sa sabiranjem ili ne.