



----- !11000001! -----	koda ukaza ADDL3
!00000010! -----	opis prvega operanda (#2)
!11001111! -----	opis drugega operanda z relativnim naslavljanjem
!11111110! -----	odmik
!00011010! -----	
!01010001! -----	opis tretjega operanda (R1)

Slika 3.1: Slika ukaza v pomnilniku.

Ko procesor izvrši nek ukaz, prebere naslednjega. Procesor torej prebere byte z naslova, ki je zapisan v programskem števcu in takoj poveča programski števec za 1. Ko raztolmači kodo ukaza, se odloči, ali je naslednji byte nadaljevanje tega ukaza. To je takrat, ko je koda ukaza dolga dva byta, ali ko je potreben za izvršitev ukaza še kakšen podatek. Ko procesor pričakuje podatek, prebere še en byte in spet poveča vrednost programskega števca za 1. Iz načina naslavljanja ugotovi, koliko dodatnih bytov določa ta podatek in ko jih bere sproti popravlja vrednost v programskem števcu. Ta postopek ponavlja, dokler ne prebere vseh podatkov in takrat kaže programski števec na kodo naslednjega ukaza.

3.3 UPORABA REGISTROV

Registre lahko uporabljamo na različne načine. Najpreprosteje je, če uporabimo register kot akumulator, torej da zapišemo podatek v register. Na ta način smo začasno shranili nek podatek in nam ni bilo treba za to iskati prostega pomnilnika, hkrati pa je ta podatek hitro dostopen, saj je register del procesorja in si pri naslednjem iskanju tega podatka prihranimo dostop do pomnilnika.

Namesto podatka lahko v register zapišemo naslov podatka. Zdaj je register kazalec ("pointer") na podatek. Z nekaterimi načini naslavljanja lahko avtomatično spreminjamo vrednost v registru s tem, da dostopamo do podatka. Taka uporaba je zelo učinkovita pri delu s tabelami in s skladom.

V register lahko zapišemo tudi odmik podatka od začetka neke strukture, začetek te strukture pa opišemo na poljuben način. Takemu naslavljanju pravimo indeksno, register pa uporabljamo kot indeksni register.