**Računalniška arhitektura - vaje 3:**

Naslov pomnilniške besede je **32 biten.**

**Operandi v pomnilniku:**

**Posredno registrsko naslavljanje oz. bazno naslavljanje:**

**Operande v register nalagamo s pomočjo baznega(posrednega) registra, ki hrani naslov pomnilniške lokacije tega operanda.**

**Dostop do operanda v 2 korakih:**

1. Naslov “spremenljivke” naložimo v bazni register:

ukaz ADR R1, STEV1 @ R1 dobi naslov STEV1

ADR je psevdoukaz (SUB R1, R15, #\_)

1. Uporabimo LOAD/STORE ukaz za dostop do operanda.

Branje: LDR R0, [R1] @( [ ] pomenijo naslov), R0 <- Mem32[R1]

Ali

Pisanje: STR R5, [R1] @ R5 -> Mem32[R1]

--------------------------------------------------------------------------------------------------------------------------------

**Operand:** 32b 16b 8b

**Direktiva:** .WORD .HWORD .BYTE

**LOAD:** LDR LDRH/LDRSH LDRB/LDRSB

**STORE:** STR STRH STRB

**Poravnanost:** Del. s 4 Del. s 2 Katerikoli naslov

LDRSH – signed(predznak)

Poravnanost – glede na naslove/byte

--------------------------------------------------------------------------------------------------------------------------------------

**Posredno registrsko naslavljanje oz. bazno naslavljanje s takojšnjim odmikom:**

1. Enak kot pri prvem načinu (ADR R1, STEV)
2. Podobno kot pri prvem načinu , le z odmikom

Branje: LDR R0, [R1, #4] @ R0 <- Mem32[R1 + odmik 4]

Ali

Pisanje: STR R5, [R1, #-8] @ R5 -> Mem32[R1 – odmik 8]

**Aritmetično-logični ukazi:**

**Takojšnje naslavljanje:**

**Operand se nahaja v ukazu.**

MOV R1, #128 @ V register R1 se zapiše vr. 128, # pomeni takojšnji operand, konstanto

Takojšnji operand je nepredznačeno 8-bit št., ki ga lahko še krožno pomaknemo za 2n bitov v levo, kjer je n lahko med 0 in 12.

**Neposredno registrsko naslavljanje:**

**Operand se nahaja v registru.**

Računanje z registri in prepisovanje vr. iz enega registra v drugega.

ADD R2, R7, R12 @ R2 <- R7 + R12

MOV R1, R4 @ R1 <- R4