**Računalniška arhitektura vaje 2:**

ARM = Advanced RISC Machine

Load store arhitektura

Reduciran nabor ukazov, vsi ukazi 32 bitni, vsi registri 32 bitni (4 bajti)

Veliko načinov naslavlanja

Veliko formatov ukazov

**CISC vs RISC**

Fizična realizacija CISC je zapletena (INTEL, AMD) -> zmogljivejši

RISC -> poudarek na software realizacijo

**Pomnilniška hierarhija:**

Glavni pomnilnik -> LCache(predpomnilnik) -> registry

**Programski model ARM:**

16 registrov + statusni register CPSR

2 skupini nač. Delovanja:

* Priviligirani (dovoljen dostop do CPSR)
* Nepriviligirani (dostop do CPSR ni dovoljen)

User mode: nepriviligiran način, v njem se izvajajo uporabniški programi, vidik varnosti

R0 – 12 -> splošnonamenski reg.

R13 -> stack pointer

R14 -> link register (kam se vrniti po skoku)

R15 -> program counter

CPSR: 4 biti za način delovanja, 1 bit za THUMB flag, 2 bita za interrupt pa fast interrupt, neporabljen proctor, 4 biti za flage N Z C V (signalizirajo stanje programa: N = negative(rez ukaza je neg.), Z = zero(rez. ukaza je 0), C = carry, V = overflow(prekoračeno št. bitov(V je za neg št.)))

**Programiranje v zbirnniku:**

**Simbolično opisujemo:**

* Ukaze z mnemoniki
* Registre
* Naslove
* Konstante

Prevajalnik ukaz prevede v strojno kodo, izračuna dejanske naslove in pomike, ustvari pomnilniško sliko programa.

Program v strojnem jeziku ni prenosljiv.

(na spletni učilnici je seznam ukazov)

**Ukazi:**

Vsi so 32-bitni

Rezultat je 32-bit, izjema je množenje

Aritmetično-logični ukazi so 3 operandni (ukaz, register za rezultat, konstante/registri za operacijo)

Load/store arhitektura

**Vrstica programa zgleda tako:**

**Oznaka: ukaz operandi @ komentar**

*Rutina1: add r3, r2, #1 @ povecaj stevec*

Stolpce ločimo s tabulatorji, dovoljeni so tudi presledki

**Operandi:**

8, 16 in 32 bitni ter predznačeni ali nepredznačeni pomnilniški operandi

Obvezna poravnanost ukazov in operandov (16 in 32 bitnih):

16 bitni poravnani na sodih naslovih

32 bitni poravnani na naslovih, deljivih s 4

**Oznake:**

Razumljivo simbolično poimenovanje pomnilniških lokacij ali vrstic v program.

S poimenovanjem pomnilniških lokacij dobimo “spremenljivke”

Za poimenovanje vrstic, na katere se sklicujemo pri skokih

**Direktive:**

.beseda -> direktiva (ukaz prevajalniku)

.text , .data -> označevanje kje se začnejo podatki/program

.align -> določanje vrste pomnilniških odsekov(poravnava operandov)

.space N -> rezervacija pomnilnika (rezerviraj N bajtov)

.word -> 32 bitna števila

.hword -> 16 bitna števila

.byte -> 8 bitna števila

.ascii, .asciiz -> niz znakov, (z = zero terminated)

.end -> ustavljanje ukaza

**Psevdoukazi:**

Niso strojni ukazi, ampak jih prevajalnik vanje prevede

ADR R0, STEV1 -> zapis vr. V register

Določanje pomnilniskih odsekkov:

Psevdoukaza za določanje pom. Slike sta:

.data, .text

Zgradba programa:

.text

…operandi / spremenljivke…

.align

\_\_start:

…ukazi…

\_\_end: b \_\_ end

**Rezervacija pom. Za spremenljivke:**

.text

.align @ obvezna poravnava

RADIUS: .space 4

Rezervacija prostora v pomnilniku:

BUFFER: .space 40 @ rezerviraj 40 bajtov

Oznaka buffer vstreza naslova, od katerega naprej se rezervira 40B prostora. S tem se lahko realizira tabele.

**Rezervacija prsotora z zač. Vr.:**

Stev1: .word 512, 1 ,1024