Sveučilište u Zagrebu

Fakultet elektrotehnike i računarstva

University of Zagreb

Faculty of Electrical Engineering and Computing

Arhitektura računala 1

*Computer Architecture 1*

**Laboratorijska vježba broj 1**

*Laboratory exercise no. 1*

Marina Rupe

0036483027

# Zadatak / Exercise

U memoriji od adrese 50016 nalazi se blok podataka zapisanih u 64-bitnom formatu s bitom za predznak (tj. format s bitom za predznak u dvostrukoj preciznosti). Niža riječ 64-bitnog broja nalazi se na nižim memorijskim adresama, a viša riječ na višim adresama (cijeli 64-bitni podatak zapisan je u little-endian poretku). Veličina bloka zapisana je u prvom 32-bitnom podatku na adresi 50016. Napišite program za FRISC-3 koji će podatke iz opisanog bloka pretvoriti u 16-bitni format dvojnog komplementa te zapisati u novi blok koji počinje na adresi 70016. Podaci u početnom bloku trebaju biti: -110,+210, -410,+3710, -7310. Novi blok mora biti zaključen s 16-bitnom vrijednošću 800016. Provjerite ispravnost rada Vašeg programa na gonjim testnim podacima prikazom na LCD-u.

# Rješenje / Solution

U prostor između linija kopirati asemblerski kod Vašeg rješenja uz komentare (na engleskom jeziku). Uključite i kod potreban za provjeru rješenja. Prilikom kopiranja koristite opciju Paste -> Keep text only kako bi sačuvali izvorni format.

Copy your assembly code between lines. Include code required for verification of the solution. Please comment the code. Use Paste -> Keep text only when copying to preserve original formatting.

ORG 0

MOVE 1000, SP

JP GLAVNI

ORG 100

GLAVNI MOVE 500, R1 ;address

MOVE LCDDATA, R6 ;address for printing on LCD

LOAD R0, (R1) ;load the number of data from  
 ;address stored in R1

STORE R0, (R6) save it for printing on LCD

ADD R6, 4, R6 ;increase address for printing

MOVE 700, R4 ;address for saving data

PETLJA ADD R1, 4, R1 ;the next data

LOAD R3, (R1) ;lower word

STORE R3, (R6) ;save lower word for printing on  
 ;the screen

ADD R1, 4, R1 ;move address on the higher word

LOAD R2, (R1) ;higher word

STORE R2, (R6+4) ;save higher word for printing on  
 ;the screen

ADD R6, 8, R6 ;increase address for printing

SHL R2, 1, R2 ;the sign is defined by the most  
;significant bit

JR\_C KOMPLEMENT ;if the most significant bit is 1  
;we make double complement

JP NASTAVI

KOMPLEMENT XOR R3, -1, R3

ADD R3, 1, R3

NASTAVI STOREH R3, (R4) ;save halfword - that is the data  
 ;we've got

AND R3, 0FFFF, R3 ;put first 16 bits in 0

STORE R3, (R6) ;save halfword for printing on the  
 ;screen

ADD R6, 4, R6 ;increase address for printing

ADD R4, 2, R4 ;increase address for 2B

SUB R0, 1, R0 ;decrease counter for 1

JR\_NZ PETLJA ;while counter isn't 0 – continue  
 ;the loop

MOVE 8000, R3 ;store 8000 as a mark of the end

STOREH R3, (R4)

JP PRINTDATA

HALT

ORG 500

DW 5, 1, 80000000, 2, 0, 4, 8000000, 25, 0, 49, 80000000

ORG 700

DH 0, 0, 0, 0, 0

PRINTDATA MOVE %B 011, R0

STORE R0, (%H FFFF0000) ;konfiguracija GPIO sklopa za SW  
 ;-ulazni nacin bez int

LOAD R0, (%H FFFF0004) ;ucitaj stanje SW

SHL R0, 4, R0

MOVE LCDDATA, R1

ADD R0, R1, R2

MOVE 4, R3

PRINTLOOP CALL LCDWRITE

ADD R2, 4, R2

SUB R3, 1, R3

CMP R3, 0

JP\_NE PRINTLOOP

JP PRINTDATA

LCDWRITE LOAD R0, (%H FFFFF008)

CMP R0, 0

JR\_EQ LCDWRITE

MOVE 1, R0

STOREB R0, (%H FFFFF004)

LOAD R0, (LCDCURS)

ADD R0, 1, R0

STORE R0, (LCDCURS)

CMP R0, 3

JP\_EQ ROW2

CMP R0, 6

JP\_EQ ROW1

JP NOMOVE

ROW1 MOVE %B 10000000, R0

STOREB R0, (%H FFFFF005)

MOVE 0, R0

STORE R0, (LCDCURS)

JR LCDWRITE

ROW2 MOVE %B 11000000, R0

STOREB R0, (%H FFFFF005)

JR LCDWRITE

NOMOVE LOAD R0, (R2)

STORE R0, (%H FFFFF000)

RET

LCDCURS DW %H 00000000

LCDDATA DS %D 1024

# Ispitivanje rješenja / Solution Verification

U prostor između linija objasnite kako se provjerava ispravnost rješenja (na hrvatskom ili na engleskom jeziku):

Write detailed explanation how is the solution verified:

Na početku na zaslon ispišemo koliko ima 64-bitnih podataka (tako da kasnije znamo jesu li se svi učitali). Ispisuju se redom učitani podaci (64-bitni - prvo niža riječ pa viša), a nakon svakog učitanog podatka ispiše se taj podatak pretvoren u 16-bitni podatak u format dvojnog komplementa (na zaslonu se ispiše unutar 32 bita tako da je prvih 16 bitova u 0) i tako redom za sve podatke.

Na taj način možemo vidjeti je li se zadani broj pretvorio