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

Napisati potprogram INVERZ koji računa bitovni inverz (ili zrcalni podatak) od 32-bitnog podatka. Podatak se prenosi u potprogram preko R0, a rezultat se vraća također preko R0. Primjer: Bitovni inverz broja 1110 0010 1010 11112 je broj 111101010100 01112. Napisati potprogram PALIN koji za svaki podatak zadanog bloka, korištenjem potprograma INVERZ, ispituje je li podatak bitovni palindrom, odnosno ispituje je li njegova zrcalna vrijednost jednaka originalnoj vrijednosti. Potprogram PALIN prima adresu i duljinu zadanog bloka preko stoga, a u R0 vraća ukupni broj pronađenih palindroma. Napisati glavni program koji poziva potprogram PALIN za zadani blok podataka koji počinje na adresi 50016 i ima 51210 podataka, te vraćeni ukupni broj palindroma pohranjuje na adresu 45016.

# 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 %D 10, R2 ;number of data -- R2

MOVE LCDDATA, R5 ;memory address where we put ;data for printing on LCD

STORE R2, (R5) ;save the number of data  
 ;for printing on the screen

ADD R5, 4, R5 ;increase address for

;printing on LCD

MOVE 500, R3 ;address -- R3

MOVE 0, R1 ;counter -- R1

STORE R1, (R5)

ADD R5, 4, R5 ;increase address for

;printing on LCD

PUSH R3 ;send arguments – the number of

;data and the address

PUSH R2

CALL PALIN

ADD SP, 8, SP ;remove parameters from stack

STORE R1, (450) ;store the number of palindromes

STORE R1, (R5) ;save that number for printing  
 ;on LCD

JP PRINTDATA ;print saved data on LCD

HALT

PALIN LOAD R2, (SP+4) ;load parameters from stack

LOAD R3, (SP+8)

PETLJAPRVA LOAD R0, (R3) ;load data from address

;which is saved in R3

STORE R0, (R5) ;show data on the screen

ADD R5, 4, R5 ;increase address for

;printing on LCD

MOVE R0, R4 ;copy data to register R4

CALL INVERZ

STORE R0, (R5) ;print the inverse of the data

;on the screen

ADD R5, 4, R5 ;increase address for

;printing on LCD

CMP R0, R4 ;compare the old and the new data

JR\_EQ BROJAC ;if they are the same, increase

;the counter

JP NASTAVI

BROJAC ADD R1, 1, R1 ;increase the counter

NASTAVI ADD R3, 4, R3 ;increase address for 4B

SUB R2, 1, R2 ;decrease number of data which  
;have to be tested

JR\_NZ PETLJAPRVA ;stay in loop while number of data  
;isn't 0

RET

INVERZ PUSH R1 ;save data from R1,R2,R3

PUSH R2

PUSH R3

OR R0, R0, R1 ;move data to register R1

MOVE 0, R0 ;empty the register R0;

MOVE %D 31, R3 ;counter

PETLJADRUGA MOVE 0, R2 ;set R2 in 0

SHR R1, 1, R1 ;move in the flag C the rightmost  
;bit

ADC R2, R2, R2 ;sum R2 which is 0 and carry and

;save it to R2 -- the rightmost  
;bit

ROTL R2, R3, R2 ;rotate that bit on the right place

OR R2, R0, R0 ;copy bit to R0

SUB R3, 1, R3

CMP R3, -1 ;finish when the counter R3 = -1

JR\_NE PETLJADRUGA

POP R3 ;return registers R1,R2,R3

POP R2

POP R1

RET

ORG 450

BROJ DW 0

ORG 500

POD1 DW %B 11111111111111111111111111111111

POD2 DW %B 01111111111111111111111111111110

POD3 DW %B 00000000000000000000000000000000

POD4 DW %B 00000000000000110000000000000000

POD5 DW %B 00000000000011001100000000000000

POD6 DW %B 00000000001100000011000000000000

POD7 DW %B 00000000000011111111000000000000

POD8 DW %B 00110000000011001100000000001100

POD9 DW %B 00000001110000011100000000000111

POD10 DW %B 00000000111000000000111000011100

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:

Prvo na LCD ispišemo broj podataka koje treba testirati (da znamo kasnije jesu li se svi testirali). Ispravnost rješenja provjerava se tako da se svaki učitani podatak ispiše na zaslon, za njim njegov izračunati inverz te nakon što se tako izredaju svi podaci ispiše se broj pronađenih palindroma.