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*

Dora Mlinarić 0036480184

Kristina Vučenik 0036476704

# Zadatak / Exercise

2.5.13 Subtracting and dividing numbers in double precision

Write a subprogram SUB\_DBL for subtracting numbers in double precision. Numbers in double precision are being stored in the memory like this: lower part of the number is being stored first, and then the higher part of the number. Address of the first operand (lower part) is stored in R0 register and address of the second operand (lower part) is stored in R1 register. Subtract second operand from the first one. Register R2 contains the address where the result has to be stored.

Write a subprogram DIV\_DBL which (using SUB\_DBL subprogram) divides two numbers in double precision using the method of repeated subtraction. Register R0 contains the address of the dividend (lower part) and the R1 register contains the address of the divisor(lower part). In R2 register there is an address where the result has to be stored. Modulo can be ignored.

Write the main program which calls the DIV\_DBL subprogram and divides two numbers.

# Rješenje / Solution

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.

;program for dividing 2'k numbers

`ORG 0

MOVE 1000, SP ;stack pointer initialization

MOVE L1, R0 ;adress of the lower 4 bytes of the dividend (pointer)

MOVE L2, R1 ;adress of the lower 4 bytes of the divisor (pointer)

MOVE RL, R2 ;adress of the lower 4 bytes of the result (pointer)

MOVE 0, R5 ;counter initialization

LOAD R3, (R0+4) ;load higher 4 bytes of the dividend

CMP R3, 0 ;compare them with 0, if they are positive skip

JP\_SGE NEXT1 ;if not, turn it into positive (absolute value)

LOAD R4, (R0) ;load lower 4 bytes of the dividend

SUB R4, 1, R4 ;turn it from 2-complement to positive

SBC R3, 0, R3

XOR R4, -1, R4

XOR R3, -1, R3

STORE R4, (TEMP1L) ;store higher and lower 4 bytes to the temporary location

STORE R3, (TEMP1H)

MOVE TEMP1L, R0 ;change pointer to the dividend (temporary location)

ADD R5, 1, R5 ;increase counter (how many negative operands)

NEXT1 LOAD R3, (R1+4) ;load higher 4 bytes of the divisor

CMP R3, 0 ;compare them with 0, if they are positive skip

JP\_SGE NEXT2 ;if not, turn it into positive (absolute value)

LOAD R4, (R1) ;load lower 4 bytes of the divisor

SUB R4, 1, R4 ;turn it from 2-complement to positive

SBC R3, 0, R3

XOR R4, -1, R4

XOR R3, -1, R3

STORE R4, (TEMP2L) ;store higher and lower 4 bytes to the temporary location

STORE R3, (TEMP2H)

MOVE TEMP2L, R1 ;change pointer to the divisor (temporary location)

ADD R5, 1, R5 ;increase counter (how many negative operands)

NEXT2 CALL DIV\_DBL ;call the subprogram for dividing

LOAD R3, (R2) ;load the result

LOAD R4, (R2+4)

CMP R5, 1 ;compare counter with 1, if it is equal to 1 then only one

; number was

JP\_NE NEXT3 ;negative so the result also has to be negative, otherwise it

;stays positive

XOR R3, -1, R3 ;turn it into 2-complement from absolte value

XOR R4, -1, R4

ADD R3, 1, R3

ADC R4, 0, R4

STORE R3, (R2) ;store higher and lower 4 bytes of the result to the specified ;adress

STORE R4, (R2+4)

NEXT3 MOVE LCDDATA, R6 ;lcd preview location pointer

STORE R4, (R6) ;store the result on the adress for lcd preview

STORE R3, (R6+4)

JP PRINTDATA ;go to the subpogram for lcd preview

HALT

;subpogram for the numbers subtraction

SUB\_DBL PUSH R3 ; store registers

PUSH R4

PUSH R5

LOAD R3, (R0) ;load lower 4 bytes of the minuend

LOAD R4, (R1) ;load lower 4 bytes of the subtrahend

SUB R3, R4, R5 ;subtract lower 4 bytes

STORE R5, (R2) ;store lower 4 bytes of the result

LOAD R3, (R0+4) ;load higher 4 bytes of the minuend

LOAD R4, (R1+4) ;load higher 4 bytes of the subtrahend

SBC R3, R4, R5

STORE R5, (R2+4) ;store higher part of the result

POP R5

POP R4

POP R3 ;refresh registers

RET

;subpogram for dividing

DIV\_DBL PUSH R3 ;store registers

PUSH R4

PUSH R5

MOVE -1, R3 ;counter initialization

MOVE 0, R5

LOOP ADD R3, 1, R3 ;increase counter (lower part of the result)

CMP R3, -1 ;compare if there has been an overflow in the counter

JP\_NE NEXT ;if not skip

ADD R5, 1, R5 ;if there was an overflow increase higher 4 bytes for 1, and

;set lower ones to 0

MOVE 0, R3

NEXT CALL SUB\_DBL ;call the subpogram for the numbers subtraction

MOVE R2, R0 ;new minuend is the subtraction result

LOAD R4, (R2+4) ;load higher 4 bytes of the result

ADD R4, 0, R4

JP\_P LOOP ;if negative stop

STORE R3, (R2) ;store lower part of the conter/result

STORE R5, (R2+4) ;store higher part of the conter/result

POP R5

POP R4

POP R3 ;refresh registers

RET

; memort locations with data

L1 DW 12345678 ;enter the dividend

H1 DW 87654321

L2 DW 12345678 ;enter the divisor

H2 DW 23322110

RL DW 0

RH DW 0

TEMP1L DW 0

TEMP1H DW 0

TEMP2L DW 0

TEMP2H DW 0

PRINTDATA MOVE %B 001, R0

STORE R0, (%H FFFF0000) ;GPIO configuration for SW – input without int

LOAD R0, (%H FFFF0004) ;load SW state

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 00000001, R0

STOREB R0, (%H FFFFF005)

MOVE 0, R0

STORE R0, (LCDCURS)

JR LCDWRITE

ROW2 MOVE %B 10000001, R0

STOREB R0, (%H FFFFF005)

JR LCDWRITE

NOMOVE LOAD R0, (R2)

STORE R0, (%H FFFFF000)

RET

LCDCURS DW %H 00000000

LCDDATA `DS %D1024

# Ispitivanje rješenja / Solution Verification

Write detailed explanation how is the solution verified:

Using the LCD preview we can see the result of our program. Checking if it is correct is very simple, we can use any better calculator to divide/subtract our numbers and check the result.