

# *Wojskowa Akademia Techniczna im. Jarosława Dąbrowskiego*

Laboratorium z przedmiotu:  
Architektura i organizacja komputerów

## Sprawozdanie z ćwiczenia laboratoryjnego nr 5: **WinDLX – wprowadzenie**

### Spis treści

Wybrany rozkaz przesłań - .....	2
Opis rozkazu:.....	Error! Bookmark not defined.
Kod programu: .....	Error! Bookmark not defined.
Spodziewany rezultat: .....	Error! Bookmark not defined.
Wybrany rozkaz arytmetyczny/logiczny – ADD .....	2
Opis rozkazu:.....	3
Kod programu: .....	3
Spodziewany rezultat: .....	3
Wykonanie programu:.....	4
Wybrany rozkaz skoków – .....	4
Wybrany rozkaz zmiennoprzecinkowy - .....	5

## Wybrany rozkaz przesłań - LW

Opis rozkazu:

Lw R, A

Rozaz wpisuje w rejestr R wartość A (na podstawie rejestru lub zmiennej)

Kod programu:

.data

l1: .word 5

l2: .word 2500000000

.text

lw r1, l1

lw r2, l2

trap 0

Dane wejściowe to:

- l1 = 5
- l2 = 2500000000

Spodziewany rezultat:

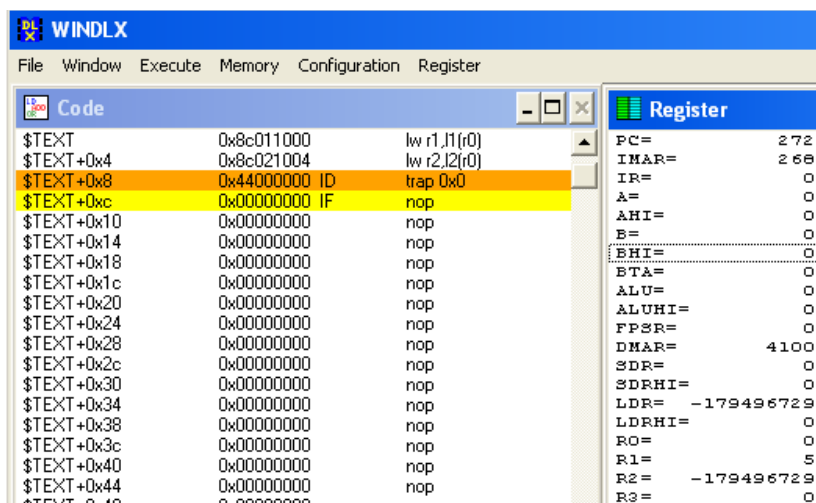
Zawartość rejestru r1:

- Zawiera przypisaną mu wartość zmiennej l1, czyli 5

Zawartość rejestru r2:

- Ponieważ zmienna wykracza poza zakres int, do rejestru zostanie przypisana wartość ujemna

Wykonanie programu:



The screenshot shows the WINDLX debugger interface. The 'Code' window on the left displays assembly instructions with their addresses and hex values. The instruction at address 0x44000000 is highlighted in yellow and reads 'trap 0x0'. The 'Register' window on the right shows the current state of the registers. The PC register is at 272, IMAR at 268, and R2 is at -179496729, which is the negative value of 2500000000.

Code	Register
\$TEXT 0x8c011000 lw r1,l1(r0)	PC= 272
\$TEXT+0x4 0x8c021004 lw r2,l2(r0)	IMAR= 268
\$TEXT+0x8 0x44000000 ID trap 0x0	IR= 0
\$TEXT+0xc 0x00000000 IF nop	A= 0
\$TEXT+0x10 0x00000000 nop	AHI= 0
\$TEXT+0x14 0x00000000 nop	E= 0
\$TEXT+0x18 0x00000000 nop	BHI= 0
\$TEXT+0x1c 0x00000000 nop	ETA= 0
\$TEXT+0x20 0x00000000 nop	ALU= 0
\$TEXT+0x24 0x00000000 nop	ALUHI= 0
\$TEXT+0x28 0x00000000 nop	FPSR= 0
\$TEXT+0x2c 0x00000000 nop	DMAR= 4100
\$TEXT+0x30 0x00000000 nop	SDR= 0
\$TEXT+0x34 0x00000000 nop	SDRHI= 0
\$TEXT+0x38 0x00000000 nop	LDR= -179496729
\$TEXT+0x3c 0x00000000 nop	LDRHI= 0
\$TEXT+0x40 0x00000000 nop	RO= 0
\$TEXT+0x44 0x00000000 nop	R1= 5
\$TEXT+0x48 0x00000000 nop	R2= -179496729
\$TEXT+0x4c 0x00000000 nop	R3= 0

## Wybrany rozkaz arytmetyczny/logiczny – ADD

Opis rozkazu:

ADD R, A, B

Rozkaz dodaje wartości A i B (na podstawie zmiennych lub rejestrów) zwraca wynik na rejestr R

Kod programu:

.data

I2: .word 5

I3: .word 10

I5: .word 1234567890

I6: .word 1987654320

.text

lw r2, I2

lw r3, I3

lw r5, I5

lw r6, I6

add r1, r2, r3

add r4, r5, r6

trap 0

Dane wejściowe to:

- I2 = 5
- I3 = 10
- I5 = 1234567890
- I6 = 1987654320

Spodziewany rezultat:

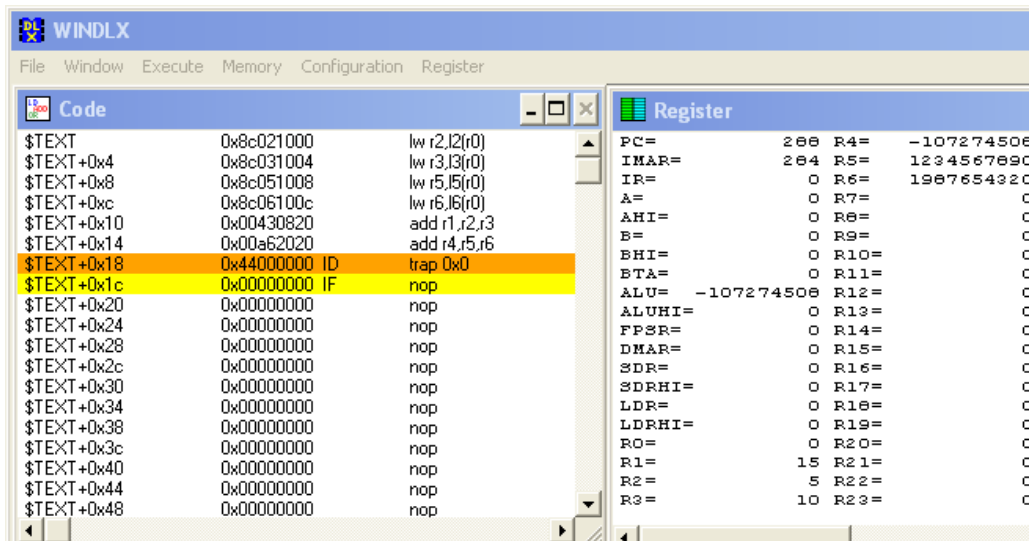
Zawartość rejestru r1:

- Zawiera sumę rejestrów r2 i r3, w których znajdują się wartości kolejnych zmiennych I2 i I3, 5 + 10 = 15, czyli r1 = 15

Zawartość rejestru r4:

- Zawiera sumę rejestrów r5 i r6, w których znajdują się wartości kolejnych zmiennych I5 i I6, 1234567890 + 1987654320 = 3 222 222 210, wartość przekracza zakres int, więc nastąpi przeładowanie, a w r4 znajdzie się ujemna wartość

Wykonanie programu:



## Wybrany rozkaz skoków – BNEZ

Opis rozkazu:

BNEZ R, label

Rozkaz sprawdza czy rejestr R jest równy 1 i jeśli tak, wykonuje przeskoczenie do etykiety „label”

Kod programu:

.data

false: .word 0

true: .word 1

l: .word 3

.text

lw r1, false

lw r2, true

bnez r1, next

lw r3, l

next:

    bnez r2, finish

    lw r4, l

finish:

    trap 0

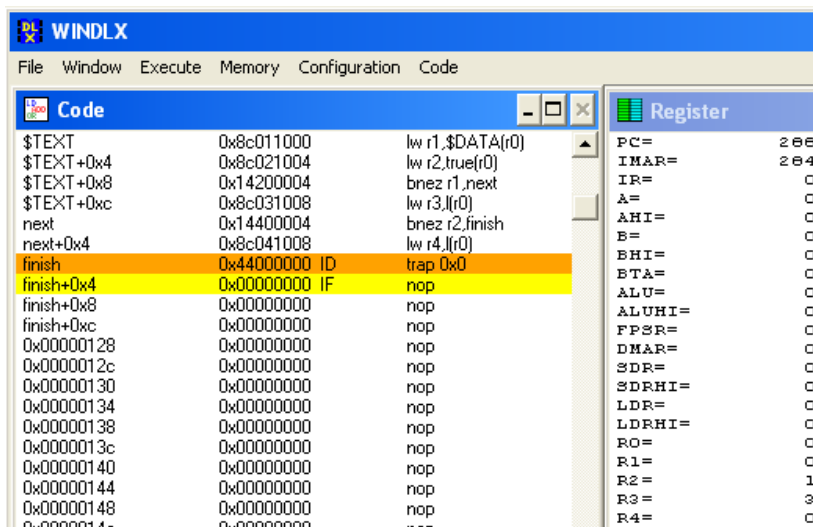
Dane wejściowe to:

- false = 0
- true = 1
- I = 3

Spodziewany rezultat:

- Instrukcja bnez wywołana po raz pierwszy nie wykona przeskoku do etykiety „next” (przez co instrukcja „lw r3, I” wykona się)
- Instrukcja bnez wywołana po raz drugi wykona przeskok do etykiety „finish” (przez co instrukcja „lw r4, I” nie wykona się)
- W związku z tym na końcu rejestr r3 otrzyma wartość, natomiast r4 jej nie otrzyma

Wykonanie programu:



## Wybrany rozkaz zmiennoprzecinkowy - LEF

Opis rozkazu:

Lef A, B

Rozkaz sprawdza, czy wartość rejestru A jest mniejsza lub równa wartości rejestru B, jeśli prawda to ustawia wartość rejestru fpsr na 1

Kod programu:

.data

I1: .float 5

I2: .float 8

I3: .float 3

.text

If f1, I1

If f2, l2

check:

lef f1, f2

bfpt finish1

If f3, l3

finish1:

If f4, l3

check2:

lef f2, f1

bfpt finish2

If f5, l3

finish2:

If f6, l3

trap 0

Dane wejściowe to:

- l1 = 5
- l2 = 8
- l3 = 3

Spodziewany rezultat:

- Pierwsze wywołanie lef będzie spełnione, więc przejdzie od razu do etykiety finish2 (If f3, l3 nie wykona się)
- Drugie wywołanie lef nie będzie spełnione, więc nie przejdzie od razu do etykiety finish2 (If f3, l3 wykona się)
- Na końcu programu f4, f5, f6 otrzymają wartość, natomiast f3 jej nie otrzyma

Wykonanie programu:

**WINDLX**

File Window Execute Memory Configuration Register

**Code**

\$TEXT	0x98011000	If f1,f1(r0)
\$TEXT+0x4	0x98021004	If f2,f2(r0)
check	0x04220014	lef f1,f2
check+0x4	0x18000004	bfpt finish1
check+0x8	0x98031008	If f3,f3(r0)
finish1	0x98041008	If f4,f3(r0)
check2	0x04410014	lef f2,f1
check2+0x4	0x18000004	bfpt finish2
check2+0x8	0x98051008	If f5,f3(r0)
finish2	0x98061008	If f6,f3(r0)
0x00000128	0x44000000	ID trap 0x0
0x0000012c	0x00000000	IF nop
0x00000130	0x00000000	nop
0x00000134	0x00000000	nop

**Register**

PC=	304	R29=	0
IMAR=	300	R30=	0
IR=	0	R31=	0
A=	0	F0=	0
AHI=	0	F1=	5
B=	0	F2=	8
BHI=	0	F3=	0
BTA=	0	F4=	3
ALU=	0	F5=	3
ALUHI=	0	F6=	3
FPSP=	0	F7=	0
DMAR=	4104	F8=	0
SDR=	0	F9=	0
SDRHI=	0	F10=	0
UDR=	1077936128	F11=	0