Laboratorium 01 WDWK
Obliczanie wielomianu z użyciem asemblera

Grzegorz Wszoła

Marzec 2025

Spis treści

1 Opis i cel zadania

Zadanie polegało na obliczeniu wielomianu n^3+3n^2-2n przy użyciu asemblera. Aby ułatwić zadanie zamiast liczyć n^3 można policzyć $n*n*n$ co jest dużo prostszym zadaniu w asemblerze. Do zadania została wybrana architektura 32-bitowa i składnia języka AT&T.

1.1 Użyte komendy

- mov - kopiowanie danych np. mov $1, %eax
- imul - mnożenie ze sobą 2 liczb np. imul %eax, %eax
- add - dodawanie ze sobą 2 liczb np. add $4, %eax
- sub - odejmowanie od siebie 2 liczb np. sub %eax, $4
- printf - funkcja standardowa libc
- scanf - funkcja standardowa libc

2 Opis kodu

2.1 Tworzenie "zmiennych" i deklaracja funkcji

```
.section .data
    nsq:             .long 0
    format_out:      .asciz "Wynik: %d\n"
    format_in:       .asciz "%d"
    insert_text:     .asciz "Podaj n: "
    overflow_message: .asciz "Overflow occurred\n"
    max_value:       .long 2147483647  # Max 32-bit signed integer value

.section .bss
    n:      .long 0  # User input
    result: .long 0  # Storage for result

.section .text
    .global main
    .extern printf
    .extern scanf
    .extern fflush
    .extern stdout
```

2.2 Stworzenie funkcji łapiącej overflow

```
    overflow_error:
    # Handle overflow error and display an error message
    pushl $overflow_message
    call printf
    popl %eax

    # FFlush to show the message
    pushl stdout    # Push address of stdout
    call fflush
    popl %eax

    # Exit program with an error code
    movl $1, %ebx
    movl $1, %eax
    int $0x80
```

2.3 Wypisanie w konsoli pierwszego tekstu i zebranie danych od użytkownika

```
main:
# Print prompt message asking for user input
pushl $insert_text
call printf
popl %eax

# Get user input for n
pushl $n
pushl $format_in
call scanf
popl %eax
popl %eax

# Check if the input exceeds max_value
movl n, %eax
movl $max_value, %ebx
cmpl %ebx, %eax
jg overflow_error
```

2.4 Obliczanie równania

```
# Calculate n*n*n (n^3)
movl n, %eax
imull %eax, %eax
jo overflow_error
movl %eax, nsq
movl %eax, result
imull n, %eax
jo overflow_error
movl %eax, result

# Calculate result + 3*n*n (3*n^2)
movl nsq, %eax
imull $3, %eax
jo overflow_error
addl %eax, result
jo overflow_error

# Calculate result - 2*n
movl n, %eax
imull $2, %eax
jo overflow_error
movl result, %ecx
subl %eax, %ecx
jo overflow_error
movl %ecx, result
```

2.5 Wypisanie wyniku i wyjście z aplikacji

```
    # Print the result
    pushl result
    pushl $format_out
    call printf
    popl %eax

    # Exit program normally
    movl $0, %ebx
    movl $1, %eax
    int $0x80
```

3 Do kompilacji użyte zostało CMake aby przyspieszyć wpisywanie komend

```
    cmake_minimum_required(VERSION 3.10)
    project(AssemblerApp VERSION 1.0 LANGUAGES C)

    set(ASM_SOURCE ${CMAKE_CURRENT_SOURCE_DIR}/src/LabWDWK.asm)
    set(EXECUTABLE_NAME Lab01)
    set(ASM_FLAGS "--32")
    set(LINKER_FLAGS "-m32")
    set(CMAKE_ASM_COMPILER as)
    set(OBJ_FILE ${CMAKE_BINARY_DIR}/LabWDWK.o)
    set(EXEC_FILE ${CMAKE_BINARY_DIR}/${EXECUTABLE_NAME})

    message(STATUS "ASM source file: ${ASM_SOURCE}")

    add_custom_command(
        OUTPUT ${EXEC_FILE}
        COMMAND ${CMAKE_ASM_COMPILER} ${ASM_FLAGS} -o ${OBJ_FILE} ${ASM_SOURCE}
        COMMAND gcc ${LINKER_FLAGS} -o ${EXEC_FILE} ${OBJ_FILE}
        DEPENDS ${ASM_SOURCE}
    )
    add_custom_target(run ALL DEPENDS ${EXEC_FILE})
```

Do kompilacji użyty została komenda
as –32 -o {nazwa pliku wyjściowego}.o {nazwa pliku źródłowego}.asm
Do linkowania
gcc -m32 -o {nazwa pliku executable} {nazwa pliku}.o

4 Sprawdzenie wyników

Lista wyników aplikacji dla kilku losowych zmiennych podanych przez użytkownika

```
Podaj n: 2
Wynik: 16
```

Rysunek 1: Obliczony wynik dla 2

```
Podaj n: 6
Wynik: 312
```

Rysunek 2: Obliczony wynik dla 6

```
Podaj n: 10
Wynik: 1280
```

Rysunek 3: Obliczony wynik dla 10

```
Podaj n: -5
Wynik: -40
```

Rysunek 4: Obliczony wynik dla -5

Sprawdzenie obliczeń

- $2^3 + 3 * 2^2 - 2 * 2 = 8 + 12 - 4 = 16$
- $6^3 + 3 * 6^2 - 2 * 6 = 216 + 108 - 12 = 312$
- $10^3 + 3 * 10^2 - 2 * 10 = 1000 + 300 - 20 = 1280$
- $(-5)^3 + 3 * (-5)^2 - 2 * (-5) = -125 + 75 + 10 = -40$