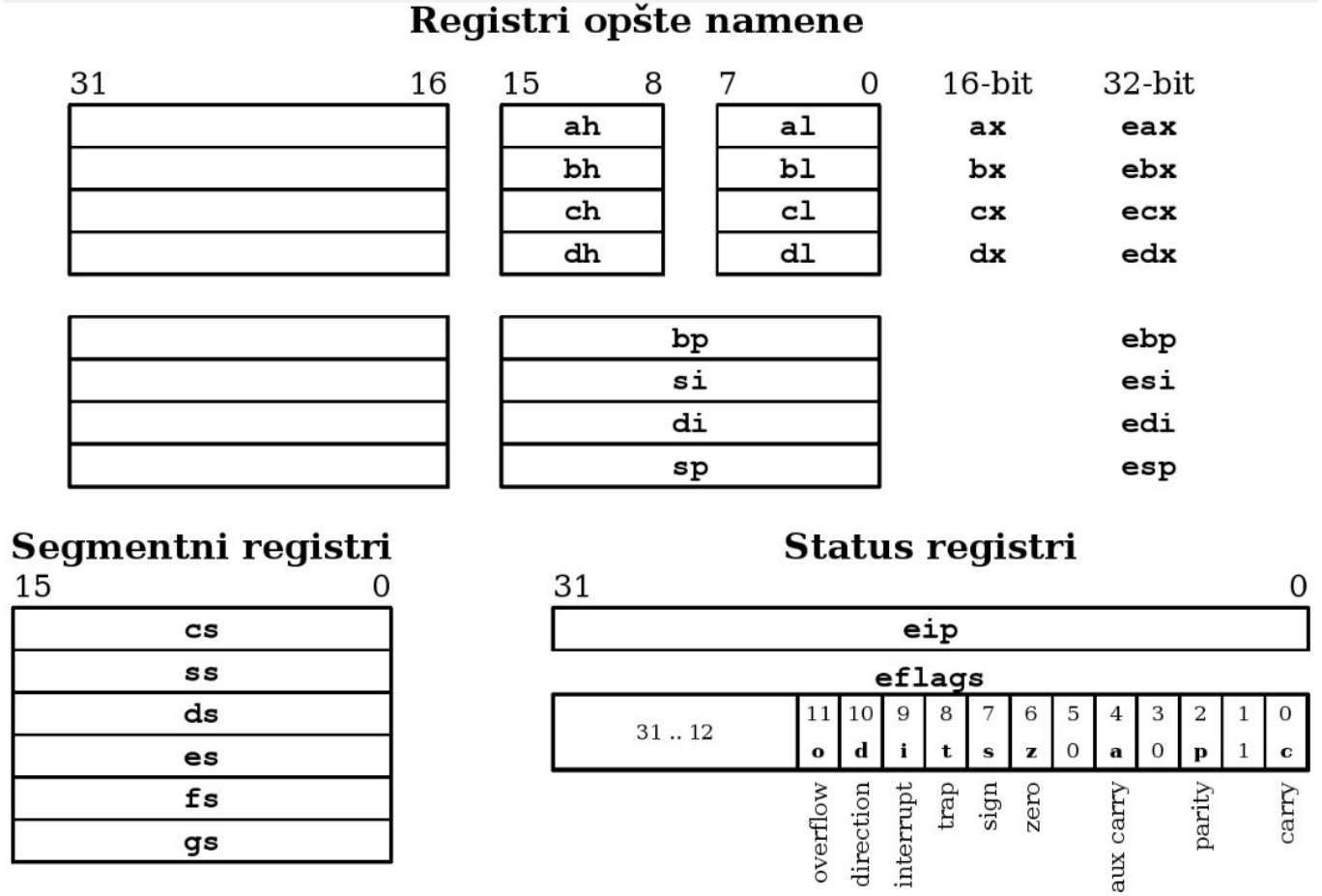
Arhitektura računara

Asemblerski programi

Zašto asembler?

- Sistemski softver se ne može pisati samo u jezicima visokog nivoa.
- Iskorišćenje potencijala procesora
  - limiti frekvencije i paralelizacije
- Kako na niskom nivou izgledaju iskazi jezika visokog nivoa.

Osnovni registri procesora Intel 80386



Indikatori (*flags*; flegovi)

- **c** (*carry*) - indikator prekoračenja za neoznačene brojeve
- **o** (*overflow*) - indikator prekoračenja za označene brojeve
- **z** (*zero*) - indikator nule
- **s** (*sign*) - indikator znaka

AT&T sintaksa

- Osnovni format:
  mnemonik izvor, odredište
- Registri: prefiks “**%**”
- Neposredni operandi: prefiks “**$**”
- Veličina operanda:
  - mnemonik ima sufiks: “**b**”, “**w**” ili “**l**”
- Labela: **ime labele** + “**:**”

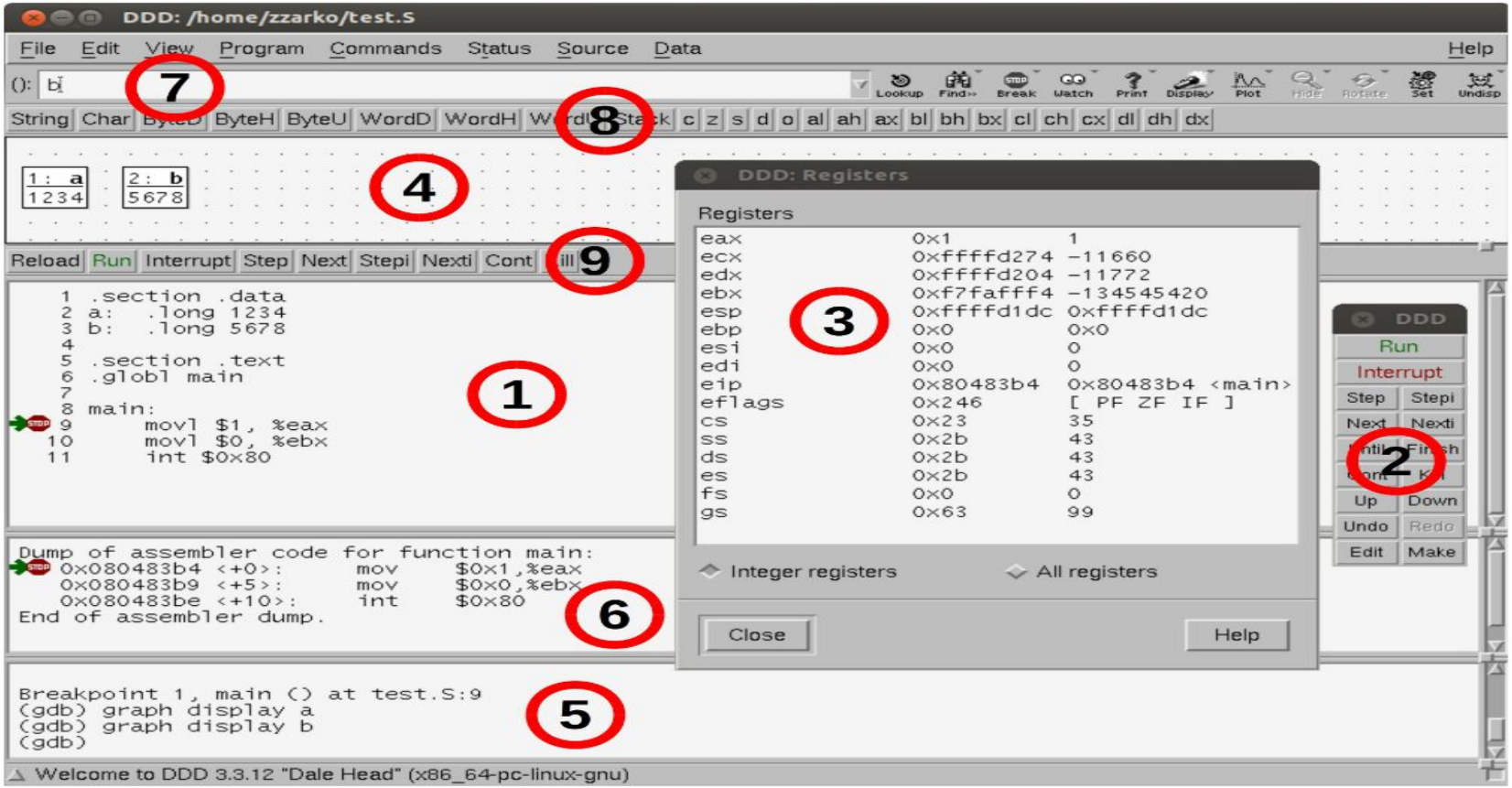
Komandna linija

- Svrha je komunikacija sa korisnikom pomoću komandi.
- Komandni prompt
  - označava da je sistem spreman:
    user@computer:~$
- Istorija komandi i kompletiranje naziva.
- Direktorijumi:
  - kreiranje: **mkdir**
  - promena tekućeg direktorijuma: **cd**
- Pokretanje programa iz tekućeg direktorijuma: **./naziv_programa**

Kompajliranje

- Osnovni oblik:
  gcc primer.c
  gcc primer.S
- Oblik koji će se ovde koristiti:
  gcc -m32 -g -o primer primer.c
  gcc -m32 -g -o primer primer.S
- Opcija “**-g**” - upisuju se informacije neophodne dibageru
- Opcija “**-o <ime>**” - zadaje se ime izlaznog fajla
  - ako se ime ne zada, podrazumevano ime je “**a.out**”
- Opcija “**-m32**” - 32-bitni program

DDD dibager


DDD: /home/zzarko/test.S
File Edit View Program Commands Status Source Data Help
String Char ByteD ByteH ByteU WordD WordH WordU Stack c z s d o al ah ax bl bh bx cl ch cx dl dh dx
1: a
1234
2: b
5678
Reload Run Interrupt Step Next Stepi Nexti Cont Till
1 .section .data
2 a: .long 1234
3 b: .long 5678
4
5 .section .text
6 .globl main
7
8 main:
9 movl $1, %eax
10 movl $0, %ebx
11 int $0x80
Dump of assembler code for function main:
0x080483b4 <+0>: mov $0x1,%eax
0x080483b9 <+5>: mov $0x0,%ebx
0x080483be <+10>: int $0x80
End of assembler dump.
Breakpoint 1, main () at test.S:9
(gdb) graph display a
(gdb) graph display b
(gdb)
DDD: Registers
Registers
eax 0x1 1
ecx 0xffffd274 -11660
edx 0xffffd204 -11772
ebx 0xf7fafff4 -134545420
esp 0xffffd1dc 0xffffd1dc
ebp 0x0 0x0
esi 0x0 0
edi 0x0 0
eip 0x80483b4 0x80483b4 <main>
eflags 0x246 [PF ZF IF]
cs 0x23 35
ss 0x2b 43
ds 0x2b 43
es 0x2b 43
fs 0x0 0
gs 0x63 99
Integer registers
All registers
Close
Help
DDD
Run
Interrupt
Step Stepi
Next Nexti
Up Down
Undo Redo
Edit Make
Welcome to DDD 3.3.12 "Dale Head" (x86_64-pc-linux-gnu)
1
2
3
4
5
6
7
8
9

Kostur programa

```
# program radi to i to
# autor: Pera Peric, RA15/2023
.section .data
.section .text
.globl main
main:

…

kraj:
    movl $1, %eax
    movl $0, %ebx
    int $0x80
```

Pojedine naredbe

- **mov** - prebaci izvorni operand u odredišni
- **sub -** oduzima izvorni operand od odredišnog i rezultat smešta u odredišni
- **cmp** - poredi izvorni i odredišni operand
  - samo ažurira indikatore
- **je**, **ja**, **jmp -** skokovi
  - do skoka **je** (**jump if equal**) dolazi ako je **z=1**
  - do skoka **ja** (**jump if above**) dolazi ako je **c=0** i **z=0**
- **jmp -** bezuslovni skok

C program

```
#include <stdio.h>



int main(void) {
    int a, b;

    a = 5;
    b = 3;

    b += a;

    return 0;
}
```

Asemblerski program

```
.section .data
.section .text
.globl main

main:


    movl $5, %eax
    movl $3, %ebx

    addl %eax, %ebx

kraj:
    movl $1, %eax
    movl $0, %ebx
    int $0x80
```

Izračunavanje NZD: algoritam

```
a = 12;
b = 8;
while (a != b) {
    if (a > b)
        a -= b;
    else
        b -= a;
}
```

1. Smesti vrednost 12 u promenljivu a
2. Smesti vrednost 8 u promenljivu b
3. Uporedi a i b
   - ako su jednaki, pređi na korak 9
4. Uporedi a i b
   - ako je a veće od b, pređi na korak 7
   - inače, pređi na sledeći korak
5. Oduzmi a od b i rezultat smesti u b
6. Pređi na korak 3
7. Oduzmi b od a i rezultat smesti u a
8. Pređi na korak 3
9. Kraj algoritma

Izračunavanje NZD: asembler

```
a = 12;
b = 8;
while (a != b) {
    if (a > b)
        a -= b;
    else
        b -= a;
}
```

```
movl $12, %eax
movl $8, %ebx
uporedi:
    cmpl %ebx, %eax
    je kraj
    cmpl %ebx, %eax
    ja vece
    subl %eax, %ebx
    jmp uporedi
vece:
    subl %ebx, %eax
    jmp uporedi
kraj:
    movl $1, %eax
    movl $0, %ebx
    int $0x80
```