

# Introduction to 8086 Assembly

## Lecture 6

Working with memory

# Why using memory?

- Registers are limited in number and size
- Program data
- Program code
- etc.



# The data segment



K. N. Toosi  
University of Technology

```
segment .data
```

```
    dd 1234
```

```
    dw 13
```

```
    db -123
```

# The data segment



K. N. Toosi  
University of Technology

```
segment .data
```

```
    dd 1234
```

```
    dw 13
```

```
    db -123
```

But how to access data?

# How to access data? Labels!



K. N. Toosi  
University of Technology

```
segment .data
```

```
l1: dd 1234
```

```
dw 13
```

```
db -123
```

# How to access data?



```
segment .data
```

```
memory1.asm
```

```
l1: dd 1234
```

```
segment .text
```

```
global asm_main
```

```
asm_main:
```

```
enter 0,0
```

```
pusha
```

```
mov eax, l1
```

```
call print_int
```

```
call print_nl
```

```
mov eax, [l1]
```

```
call print_int
```

```
call print_nl
```

# Reading data from memory



K. N. Toosi  
University of Technology

```
segment .data
```

```
l1: dd 1234
```

```
segment .text
```

```
global asm_main
```

```
asm_main:
```

```
enter 0,0
```

```
pusha
```

```
mov eax, l1
```

```
call print_int
```

```
call print_nl
```

```
mov eax, [l1]
```

```
call print_int
```

```
call print_nl
```

```
b.nasihatkon@kntu:lecture6$ ./run.sh memory1  
134520872  
1234
```

# Data labels vs. Code labels



```
segment .data
```

```
memory2.asm
```

```
l1: dd 1234
```

```
segment .text
```

```
global asm_main
```

```
asm_main:
```

```
enter 0,0
```

```
pusha
```

```
mov eax, asm_main
```

```
call print_int
```

```
call print_nl
```

```
mov eax, [asm_main]
```

```
call print_int
```

```
call print_nl
```



# Data labels vs. Code labels



K. N. Toosi  
University of Technology

```
segment .data
```

```
memory2.asm
```

```
l1: dd 1234
```

```
segment .text
```

```
global asm_main
```

```
asm_main:
```

```
enter 0,0
```

```
pusha
```

```
mov eax, asm_main
```

```
call print_int
```

```
call print_nl
```

```
mov eax, [asm_main]
```

```
call print_int
```

```
call print_nl
```

```
b.nasihatkon@kntu:lecture6$ ./run.sh memory2  
134513872  
200
```

# Putting data in memory



```
11:      db    123
12:      dw    1000
13:      db    11010b
14:      db    12o
16:      dd    1A92h
17:      dd    0x1A92
18:      db    'A'
19:      db    "B"
```

# Putting data in memory



```
11:      db    123
12:      dw    1000
13:      db    11010b
14:      db    12o
16:      dd    1A92h
17:      dd    0x1A92
18:      db    'A'
19:      db    "B"
```



**data size (not data type!)**

# Putting data in memory



```
11:    db    123
12:    dw    1000
13:    db    11010b
14:    db    12o
16:    dd    1A92h
17:    dd    0x1A92
18:    db    'A'
19:    db    "B"
```

 data size (not data type!)

```
db    1 byte
dw    2 bytes
dd    4 bytes
dq    8 bytes
dt    10 bytes
```

# Putting data in memory



```
11:      db   123
12:      dw   1000
13:      db   11010b
14:      db   12o
16:      dd   1A92h
17:      dd   0x1A92
18:      db   'A'
19:      db   "B"
```



**data size**

```
mov  eax, 11
call print_int
call print_nl

mov  eax, 12
call print_int
call print_nl

mov  eax, 13
call print_int
call print_nl

mov  al, [18]
call print_char
call print_nl
```

# Putting data in memory



```
11:      db   123
12:      dw   1000
13:      db   11010b
14:      db   12o
16:      dd   1A92h
17:      dd   0x1A92
18:      db   'A'
19:      db   "B"
```

  
**data size**

```
mov  eax, 11
call print_int
call print_nl
```

```
mov  eax, 12
call print_int
call print_nl
```

```
mov  eax, 13
call print_int
call print_nl
```

```
mov  al, [18]
call print_char
call print_nl
```

```
b.nasihatkon@kntu
134520872
134520873
134520875
A
```

# Definitions in the book



K. N. Toosi  
University of Technology

```
L1    db    0        ; byte labeled L1 with initial value 0
L2    dw    1000     ; word labeled L2 with initial value 1000
L3    db    110101b  ; byte initialized to binary 110101 (53 in decimal)
L4    db    12h      ; byte initialized to hex 12 (18 in decimal)
L5    db    17o      ; byte initialized to octal 17 (15 in decimal)
L6    dd    1A92h    ; double word initialized to hex 1A92
L7    resb   1        ; 1 uninitialized byte
L8    db    "A"      ; byte initialized to ASCII code for A (65)
```

**Carter, *PC Assembly Language*, 2007.**

# Putting data in memory



```
11:      db    123
12:      dw    1000
13:      db    11010b
14:      db    12o
16:      dd    1A92h
17:      dd    0x1A92
18:      db    'A'
19:      db    "B"
```



**data size**

**Where are the variables?**



# Putting data in memory



```
11:    db    123
12:    dw    1000
13:    db    11010b
14:    db    12o
16:    dd    1A92h
17:    dd    0x1A92
18:    db    'A'
19:    db    "B"
```



**data size**

## High-level languages (like C)

- Variables

## Low-level language (assembly)

- Labels (Symbols, Addresses)

# Putting data in memory



```
segment .data
```

```
l1: dd 11, 12, 13, 14, 15, 16
```

```
mov eax, [l1]  
call print_int  
call print_nl
```

```
mov eax, [l1+1]  
call print_int  
call print_nl
```

```
mov eax, [l1+2]  
call print_int  
call print_nl
```

```
mov eax, [l1+3]  
call print_int  
call print_nl
```

```
mov eax, [l1+4]  
call print_int  
call print_nl
```

# Putting data in memory



```
segment .data
```

```
l1: dd 11, 12, 13, 14, 15, 16
```

```
b.nasihatkon@kntu:lecture6$ ./run.sh memory4
11
201326592
786432
3072
12
```

```
mov eax, [l1]
call print_int
call print_nl
```

```
mov eax, [l1+1]
call print_int
call print_nl
```

```
mov eax, [l1+2]
call print_int
call print_nl
```

```
mov eax, [l1+3]
call print_int
call print_nl
```

```
mov eax, [l1+4]
call print_int
call print_nl
```

# Putting data in memory



```
segment .data
```

```
l1: dd 11, 12, 13, 14, 15, 16
```

```
l2: dd 8, 8, 8, 8, 8, 8, 8, 8, 8
```

```
l3: times 9 dd 8
```

```
l4: resd 9
```

```
l5: resw 18
```

```
l6: resb 36
```

# Argument types



**segment .data**

**l1: dd 11, 12, 13, 14, 15, 16**

**segment .text**

**mov eax, [l1]**  **memory**  
**mov [l1+4], ebx** 

**mov eax, ebx**  **register**

**mov eax, l1**  **immediate (constant)**  
**mov eax, 123** 

# Invalid mem,mem assembly commands



K. N. Toosi  
University of Technology

```
segment .data
11:      dd 11, 12, 13, 14
12:      dd 100

segment .text

mov [11], [12]
add [11], [12]
sub [11], [12]
adc [11], [12]
sbb [11], [12]
cmp [11], [12]
and [11], [12]
or  [11], [12]
xor [11], [12]
```

# Invalid mem,mem assembly commands



```
segment .data  
11:      dd 11, 12, 13, 14  
12:      dd 100
```

```
segment .text
```

```
mov [11], [12]  
add [11], [12]  
sub [11], [12]  
adc [11], [12]  
sbb [11], [12]  
cmp [11], [12]  
and [11], [12]  
or  [11], [12]  
xor [11], [12]
```

invalid!

# Operation size



```
segment .data  
11:      dd 11, 13, 14  
12:      dd 100
```

```
segment .text  
  
mov [11], eax  
add  eax, [12]
```

```
sub  eax, 44  
mov [11], 44
```



# Operation size



```
segment .data
11:      dd 11, 13, 14
12:      dd 100

segment .text

mov [11], eax
add  eax, [12]

sub  eax, 44
mov [11], 44
mov dword [11], 44
mov word [11], 44
mov byte [11], 44
```

# Assembly command formats



K. N. Toosi  
University of Technology

- List of x86 instructions
  - <http://www.felixcloutier.com/x86/>
  - <https://c9x.me/x86/>

# Storing multibyte data



K. N. Toosi  
University of Technology

```
segment .data
a:      dd      16753508

segment .text
:
:

mov     eax, 0

mov     al, [a]
call   print_int
call   print_nl
```

endianness1.asm

# Storing multibyte data



K. N. Toosi  
University of Technology

```
segment .data  
a:      dd    16753508
```

16753508

00

FF

A3

64

```
segment .text  
:  
:  
  
mov eax, 0  
  
mov al, [a]  
call print_int  
call print_nl
```

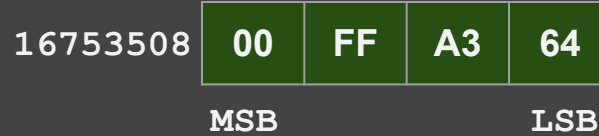
endianness1.asm

# Storing multibyte data



K. N. Toosi  
University of Technology

```
segment .data  
a:      dd      16753508
```



```
segment .text  
:  
:  
  
mov eax, 0  
  
mov al, [a]  
call print_int  
call print_nl
```

endianness1.asm

# Storing multibyte data



```
segment .data
a:      dd      16753508

segment .text
:
:

mov eax, 0

mov al, [a]
call print_int
call print_nl
```

endianness1.asm

16753508	00	FF	A3	64
	MSB		LSB	
decimal	0	255	163	100

# Storing multibyte data



```
segment .data
a:      dd      16753508

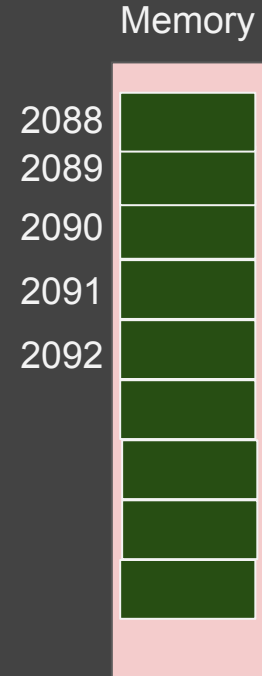
segment .text
:
:

mov eax, 0

mov al, [a]
call print_int
call print_nl
```

endianness1.asm

16753508	00	FF	A3	64
	MSB		LSB	
decimal	0	255	163	100



# Storing multibyte data



```
segment .data
a:      dd      16753508

segment .text
:
:

mov eax, 0

mov al, [a]
call print_int
call print_nl
```

endianness1.asm

16753508	00	FF	A3	64
	MSB		LSB	
decimal	0	255	163	100





# Storing multibyte data



```
segment .data
a:      dd      16753508

segment .text
:
:

mov eax, 0

mov al, [a]
call print_int
call print_nl
```

endianness1.asm

16753508	00	FF	A3	64
	MSB		LSB	
decimal	0	255	163	100



# Storing multibyte data



```
segment .data
a:      dd      16753508

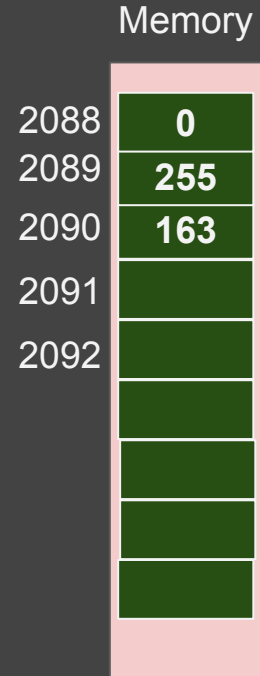
segment .text
:
:

mov eax, 0

mov al, [a]
call print_int
call print_nl
```

endianness1.asm

16753508	00	FF	A3	64
	MSB		LSB	
decimal	0	255	163	100



# Storing multibyte data



```
segment .data
a:      dd      16753508

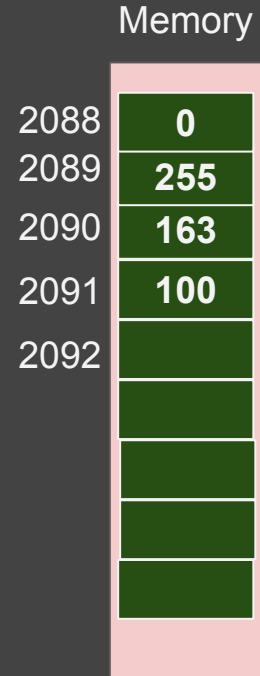
segment .text
:
:

mov eax, 0

mov al, [a]
call print_int
call print_nl
```

endianness1.asm

16753508	00	FF	A3	64
	MSB		LSB	
decimal	0	255	163	100



# Storing multibyte data



```
segment .data
a:      dd      16753508

segment .text
:
:

mov eax, 0

mov al, [a]
call print_int
call print_nl
```

endianness1.asm

16753508	00	FF	A3	64
	MSB		LSB	
decimal	0	255	163	100

Memory

2088	0
2089	255
2090	163
2091	100
2092	

```
b.nasihatkon@kntu:lecture6$ ./run.sh endianness1
100
```

# Endianness



K. N. Toosi  
University of Technology

```
segment .data
a:      dd  16753508
segment .text
:
mov  eax, 0

mov  al, [a]
call print_int
call print_nl

mov  al, [a+1]
call print_int
call print_nl

mov  al, [a+2]
call print_int
call print_nl

mov  al, [a+3]
call print_int
call print_nl
```

endianness2.asm

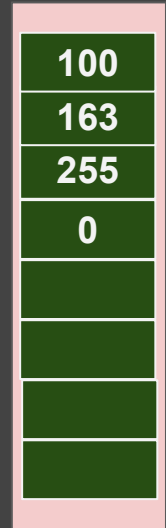
16753508	00	FF	A3	64
	MSB		LSB	
decimal	0	255	163	100

Memory



Big  
Endian

Memory



Little  
Endian  
(e.g. x86)

# Endianness



K. N. Toosi  
University of Technology

```
segment .data
a:      dd    16753508
segment .text
:
mov     eax, 0

mov     al, [a]
call   print_int
call   print_nl

mov     al, [a+1]
call   print_int
call   print_nl

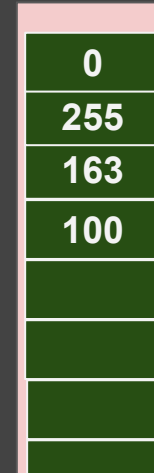
mov     al, [a+2]
call   print_int
call   print_nl

mov     al, [a+3]
call   print_int
call   print_nl
```

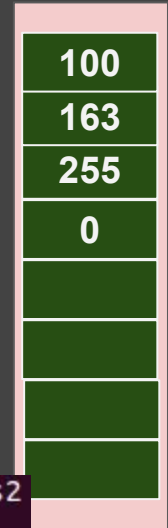
endianness2.asm

16753508	00	FF	A3	64
	MSB		LSB	
decimal	0	255	163	100

Memory



Memory



```
b.nasihatkon@kntu:lecture6$ ./run.sh endianness2
100
163
255
0
```

Little  
Endian  
(e.g. x86)

# Checking endianness in C

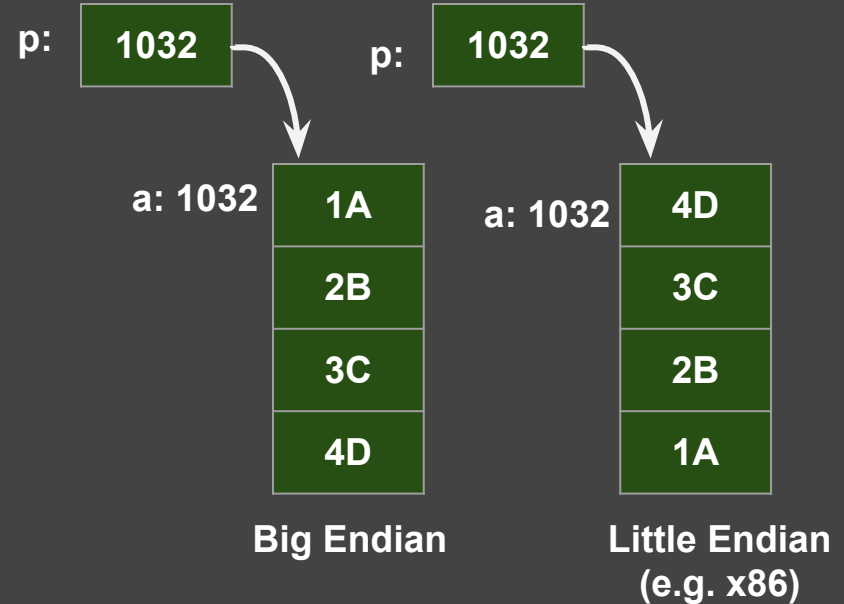


K. N. Toosi  
University of Technology

```
unsigned int a = 0x1A2B3C4D;
printf("%X\n", a);

unsigned char *p = (unsigned char *)(&a);
printf("%X\n", *p);
printf("%X\n", *(p+1));
printf("%X\n", *(p+2));
printf("%X\n", *(p+3));
```

test\_endianness.c



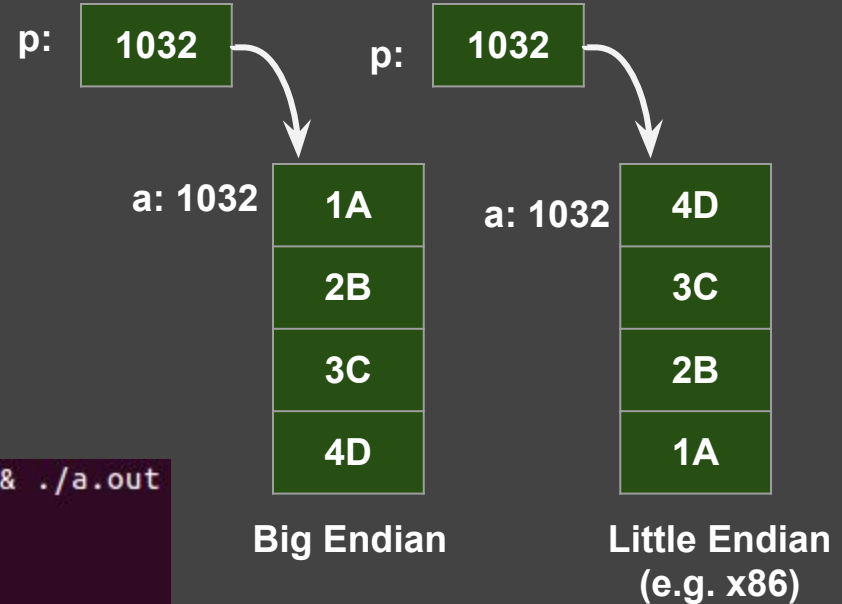
# Checking endianness in C



```
unsigned int a = 0x1A2B3C4D;
printf("%X\n", a);

unsigned char *p = (unsigned char *)(&a);
printf("%X\n", *p);
printf("%X\n", *(p+1));
printf("%X\n", *(p+2));
printf("%X\n", *(p+3));
```

```
b.nasihatkon@kntu:lecture6$ gcc test_endianness.c && ./a.out
1A2B3C4D
4D
3C
2B
1A
```





# Change endianness

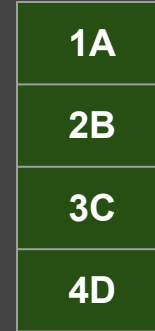


K. N. Toosi  
University of Technology

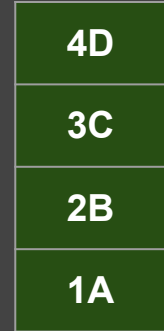
```
xchg ah, al ; 16 bit
```

```
bswap eax ; 32 bit
```

```
bswap rax ; 64 bit (x64 only)
```



Big Endian



Little Endian  
(e.g. x86)