

Introduction to 8086 Assembly

Lecture 2

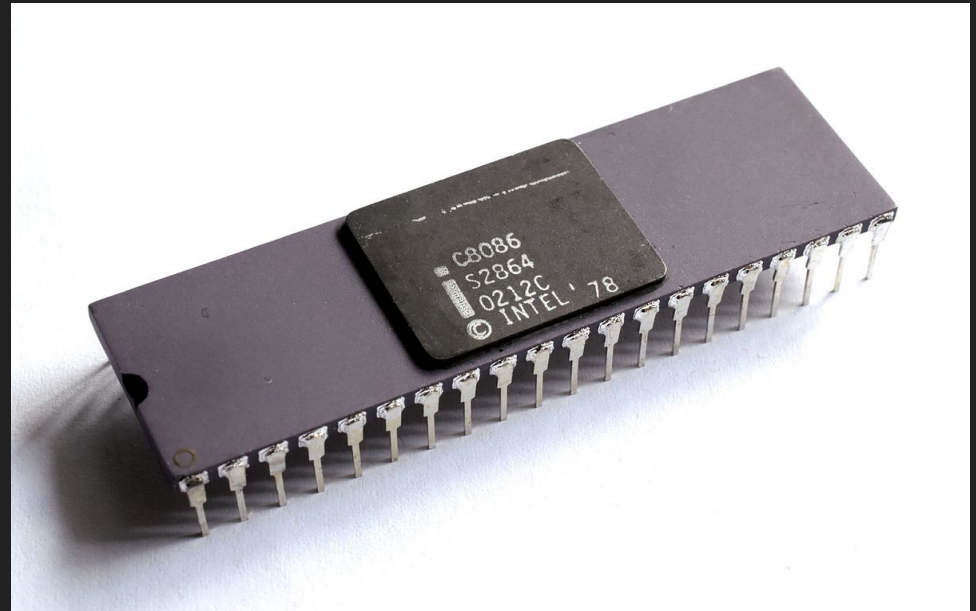
80x86 architecture, registers and basic assembly

Intel 8086 microprocessor



K. N. Toosi
University of Technology

- released in 1978
- 16 bit
- the x86 family

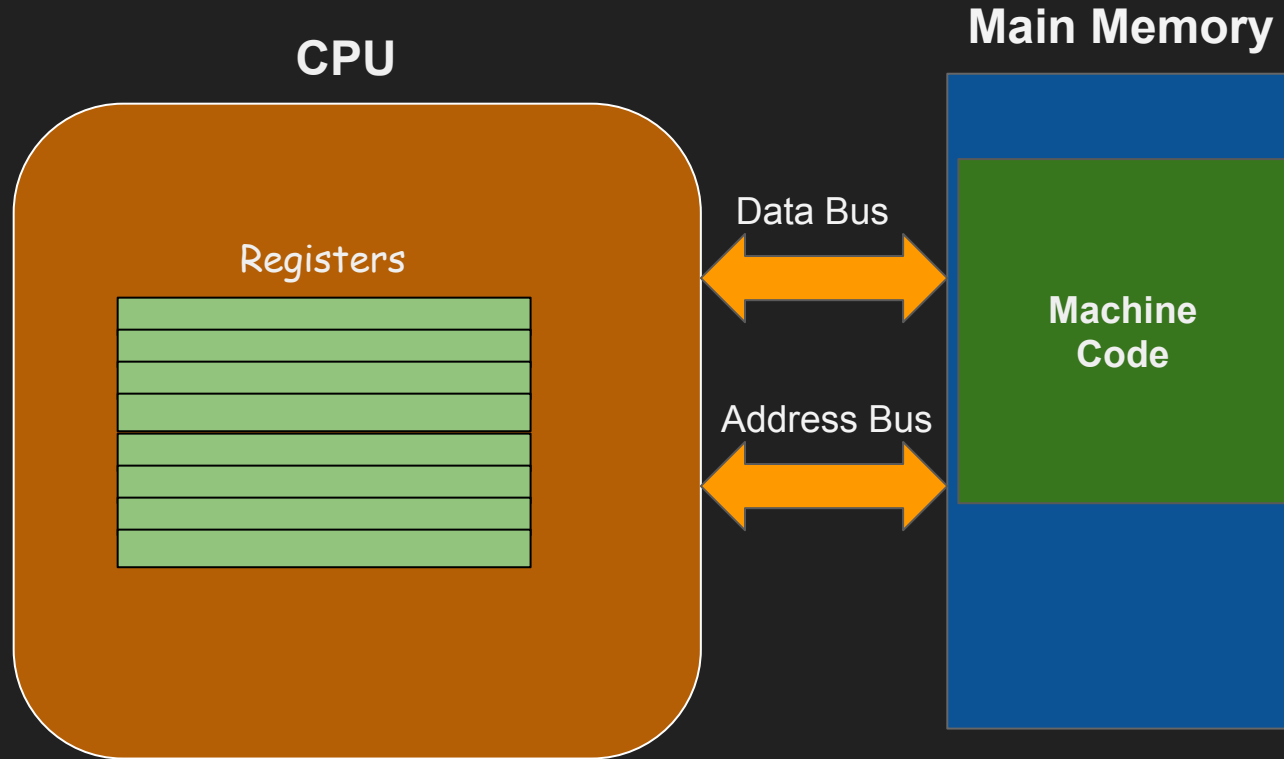


https://en.wikipedia.org/wiki/Intel_8086

CPU, Memory, instructions and data



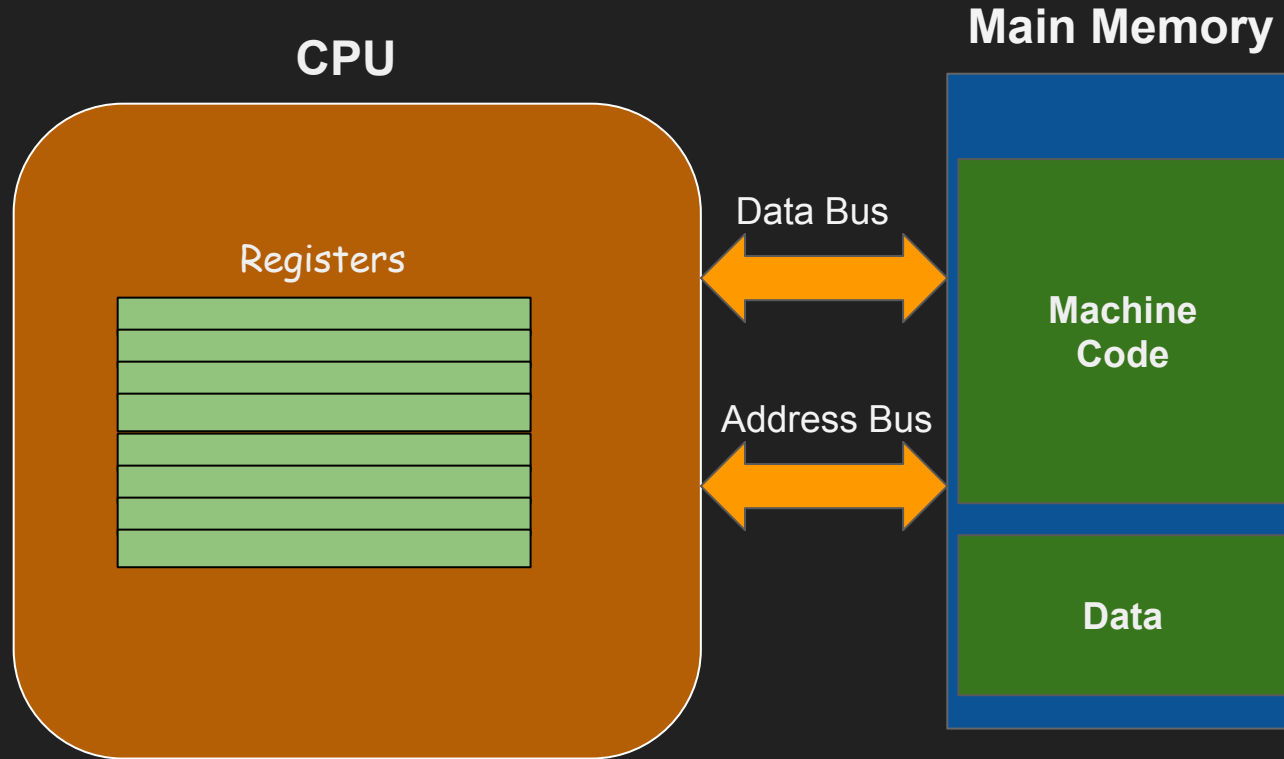
K. N. Toosi
University of Technology



CPU, Memory, instructions and data



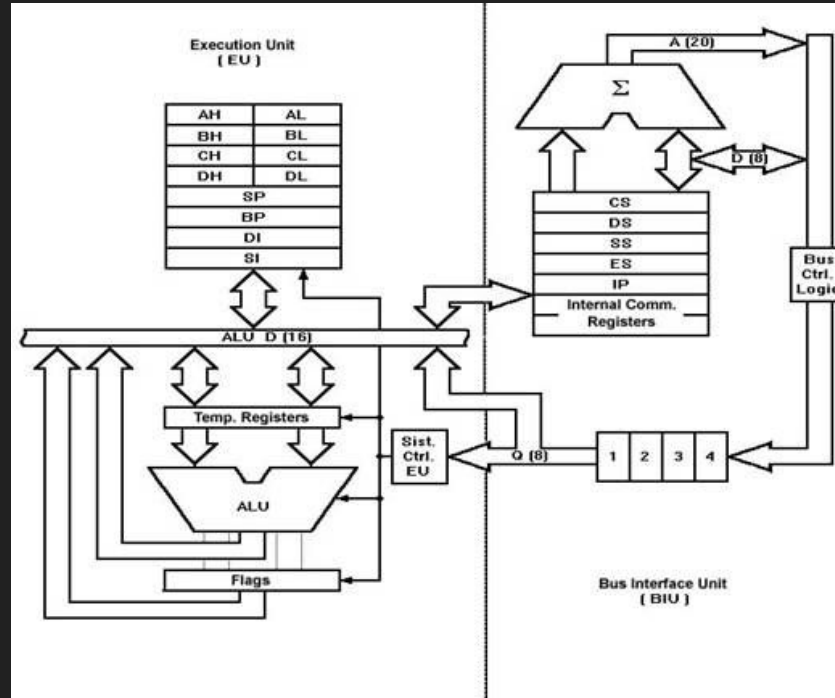
K. N. Toosi
University of Technology



Intel 8086 architecture



K. N. Toosi
University of Technology



http://www.cosc.brocku.ca/~bockusd/3p92/Local_Pages/8086_achitecture.htm



Intel 8086 registers

- 16 bit registers
- 8 bit access
 - AX,BX,CX,DX
 - e.g. AX = (AH | AL)

General Purpose Registers			
AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

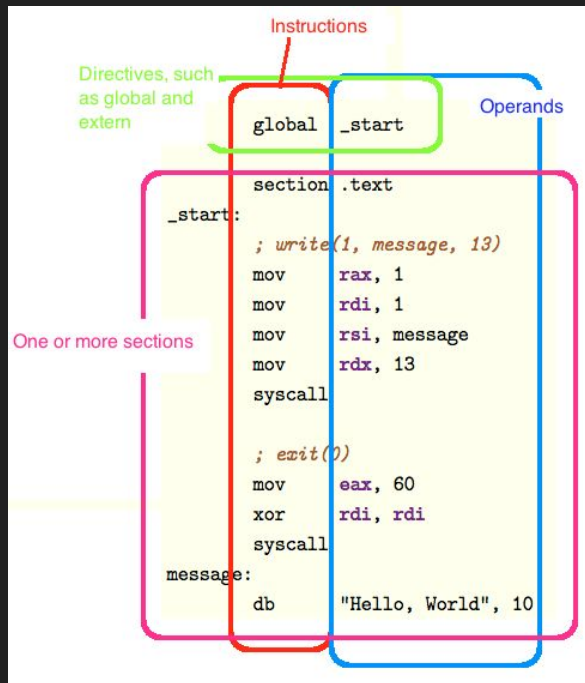
Pointer and Index Registers		
SP		Stack Pointer
BP		Base Pointer
SI		Source Index
DI		Destination Index
IP		Instruction Pointer

Segment Registers		
CS		Code Segment
DS		Data Segment
SS		Stack Segment
ES		Extra Segment

		Flags
--	--	-------



Intel 8086 assembly syntax



General Purpose Registers

AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

Pointer and Index Registers

SP		Stack Pointer
BP		Base Pointer
SI		Source Index
DI		Destination Index
IP		Instruction Pointer

Segment Registers

CS		Code Segment
DS		Data Segment
SS		Stack Segment
ES		Extra Segment

Flags



Intel 8086 assembly commands

```
mov x,y  
xchg x,y
```

```
inc x  
dec x
```

```
add x,y  
sub x,y  
neg x
```

; bitwise operators

```
and x,y  
or x,y  
xor x,y  
not x
```

General Purpose Registers

AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

Pointer and Index Registers

SP		Stack Pointer
BP		Base Pointer
SI		Source Index
DI		Destination Index
IP		Instruction Pointer

Segment Registers

CS		Code Segment
DS		Data Segment
SS		Stack Segment
ES		Extra Segment

	Flags
--	-------



Intel 8086 assembly commands

```
mov x,y
```

```
inc x
```

```
dec x
```

```
add x,y ; x <- x+y
```

```
sub x,y ; x <- x-y
```

; bitwise operators

```
and x,y ; x <- x & y
```

```
or x,y ; x <- x | y
```

```
xor x,y ; x <- x ^ y
```

x, y: register, constant, memory (address)

General Purpose Registers

AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

Pointer and Index Registers

SP		Stack Pointer
BP		Base Pointer
SI		Source Index
DI		Destination Index
IP		Instruction Pointer

Segment Registers

CS		Code Segment
DS		Data Segment
SS		Stack Segment
ES		Extra Segment

	Flags
--	-------



Intel 8086 assembly commands

assume u, v are memory locations (variables)

```
mov ax, bx
mov ax, 1
mov al, ah
mov cl, 123
mov ax, [u]
mov [v], bx
mov byte [v], 12
```

```
mov [v], [u]
```

General Purpose Registers

AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

Pointer and Index Registers

SP		Stack Pointer
BP		Base Pointer
SI		Source Index
DI		Destination Index
IP		Instruction Pointer

Segment Registers

CS		Code Segment
DS		Data Segment
SS		Stack Segment
ES		Extra Segment

	Flags
--	-------



Intel 8086 assembly commands

assume u, v are memory locations (variables)

```
mov ax, bx
mov ax, 1
mov al, ah
mov cl, 123
mov ax, [u]
mov [v], bx
mov byte [v], 12
```

~~mov [v], [u]~~

General Purpose Registers			
AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

Pointer and Index Registers		
SP		Stack Pointer
BP		Base Pointer
SI		Source Index
DI		Destination Index
IP		Instruction Pointer

Segment Registers		
CS		Code Segment
DS		Data Segment
SS		Stack Segment
ES		Extra Segment

		Flags
--	--	-------



Intel 8086 assembly commands

assume u, v are memory locations (variables)

```
mov ax, bx
mov ax, 1
mov al, ah
mov cl, 123
mov ax, [u]
mov [v], bx
mov byte [v], 12
```

~~mov [v], [u]~~

mov ax, [u]

mov [v], ax

General Purpose Registers

AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

Pointer and Index Registers

SP		Stack Pointer
BP		Base Pointer
SI		Source Index
DI		Destination Index
IP		Instruction Pointer

Segment Registers

CS		Code Segment
DS		Data Segment
SS		Stack Segment
ES		Extra Segment

Flags



Intel 8086 assembly commands

assume u, v are memory locations (variables)

```
add ax, bx
add ax, 1
add al, ah
add cl, 123
add ax, [u]
add [v], bx
add byte [v], 12
```

```
add [v], [u]
```

General Purpose Registers

AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

Pointer and Index Registers

SP		Stack Pointer
BP		Base Pointer
SI		Source Index
DI		Destination Index
IP		Instruction Pointer

Segment Registers

CS		Code Segment
DS		Data Segment
SS		Stack Segment
ES		Extra Segment

	Flags
--	-------



Intel 8086 assembly commands

assume *u, v* are memory locations (variables)

```
add ax, bx
add ax, 1
add al, ah
add cl, 123
add ax, [u]
add [v], bx
add byte [v], 12
```

~~add [v], [u]~~

General Purpose Registers

AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

Pointer and Index Registers

SP		Stack Pointer
BP		Base Pointer
SI		Source Index
DI		Destination Index
IP		Instruction Pointer

Segment Registers

CS		Code Segment
DS		Data Segment
SS		Stack Segment
ES		Extra Segment

	Flags
--	-------



Intel 8086 assembly commands

assume u, v are memory locations (variables)

```
add ax, bx
add ax, 1
add al, ah
add cl, 123
add ax, [u]
add [v], bx
add byte [v], 12
```

~~add [v], [u]~~

```
mov ax, [u]
```

```
add [v], ax
```

General Purpose Registers

AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

Pointer and Index Registers

SP		Stack Pointer
BP		Base Pointer
SI		Source Index
DI		Destination Index
IP		Instruction Pointer

Segment Registers

CS		Code Segment
DS		Data Segment
SS		Stack Segment
ES		Extra Segment

		Flags
--	--	-------



Intel 8086 assembly commands

; bx = (ax-10)*2

; ax = (ax+1)*8

; ax = (ax-1)*9

General Purpose Registers			
AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

Pointer and Index Registers		
SP		Stack Pointer
BP		Base Pointer
SI		Source Index
DI		Destination Index
IP		Instruction Pointer

Segment Registers		
CS		Code Segment
DS		Data Segment
SS		Stack Segment
ES		Extra Segment

		Flags
--	--	-------



Intel 8086 assembly commands

```
; bx = (ax-10)*2
```

```
mov bx, ax
```

```
sub bx, 10
```

```
add bx, bx
```

```
; ax = (ax+1)*8
```

```
inc ax
```

```
add ax, ax
```

```
add ax, ax
```

```
add ax, ax
```

```
; ax = (ax-1)*9
```

```
dec ax
```

```
mov si, ax
```

```
add ax, ax
```

```
add ax, ax
```

```
add ax, ax
```

```
add ax, si
```

General Purpose Registers

AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

Pointer and Index Registers

SP		Stack Pointer
BP		Base Pointer
SI		Source Index
DI		Destination Index
IP		Instruction Pointer

Segment Registers

CS		Code Segment
DS		Data Segment
SS		Stack Segment
ES		Extra Segment

	Flags
--	-------



80386 (IA-32, i386)

EAX		AH	AX	AL	Accumulator
EBX		BH	BX	BL	Base Index
ECX		CH	CX	CL	Count
EDX		DH	DX	DL	Data
ESP			SP		Stack Pointer
EBP			BP		Base Pointer
EDI			DI		Destination Index
ESI			SI		Source Index
EIP			IP		Instruction Pointer
EFLAGS			FLAGS		Flags

	CS	Code
	DS	Data
	ES	Extra
	SS	Stack
	FS	
	GS	

Figure 2-1 : The programming model of the microprocessor.

http://www.byclb.com/TR/Tutorials/microprocessors/ch2_1.htm

8086, 80286

General Purpose Registers

AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

Pointer and Index Registers

SP		Stack Pointer
BP		Base Pointer
SI		Source Index
DI		Destination Index
IP		Instruction Pointer

Segment Registers

CS		Code Segment
DS		Data Segment
SS		Stack Segment
ES		Extra Segment
		Flags



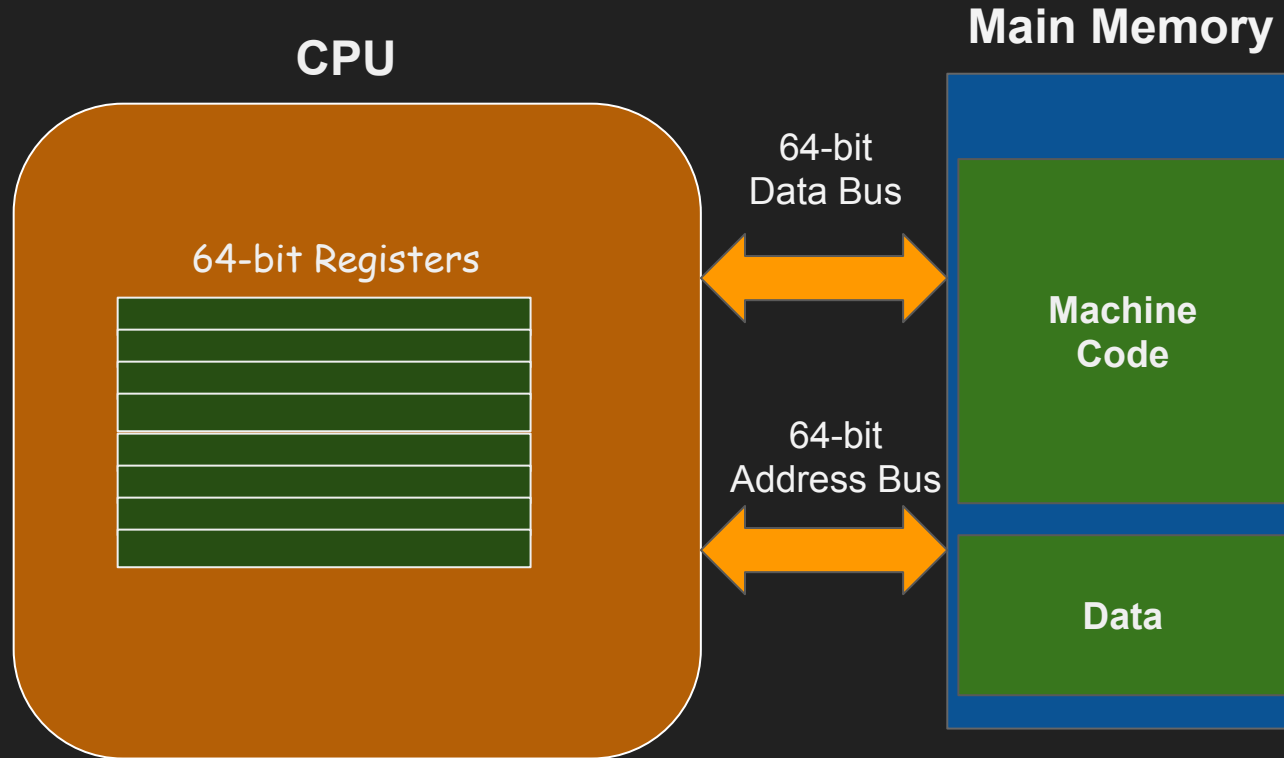
K. N. Toosi
University of Technology

<http://bucarotechelp.com/computers/architecture/86011201.asp>

64 bit x86 systems (x86-64, x64, AMD64)



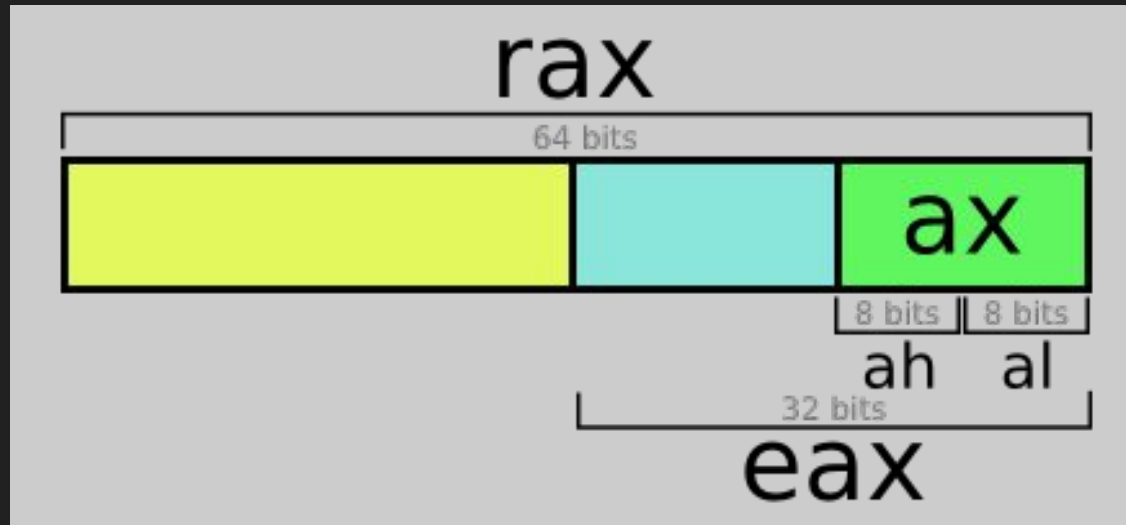
K. N. Toosi
University of Technology



x86-64 bit registers



K. N. Toosi
University of Technology



<http://nullprogram.com/blog/2015/05/15/>

x86-64 bit registers (x86-64, AMD64, Intel64, x64)



K. N. Toosi
University of Technology

