Lab 1. An Overview of Intel 8086 Architecture

Dan Bruce, David Clark and Héctor D. Menéndez

Department of Computer Science
University College London

October 9, 2017

Modified from Xeno Kovah slides of Open Security Training

Architecture

Definition (Architecture)

"The Computer Architecture is defined as the set of rules and methods that describe the functionality, organization and implementation of computer systems".

Components (I)

The architecture is divided in three main components:

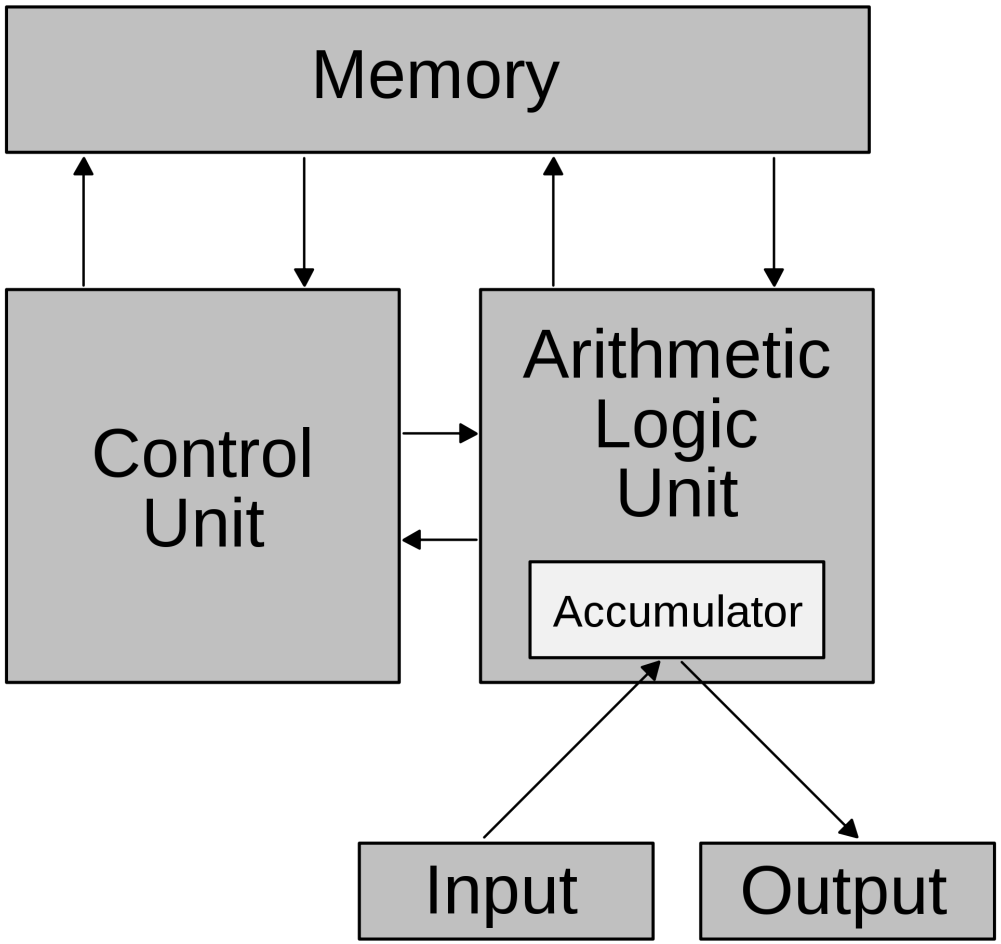
- Central Processing Unit (CPU)
- Memory
- I/O Peripheral Devices

Components (II)

The architecture is related to:

- How these components are organized (internal/external).
- How these components work.
- How these components communicate.

Von Neumman Architecture



Central Processing Unit (CPU)

Definition (Central Processing Unit)

"Electronic circuitry within a computer that carries out the instructions of a computer program by performing the basic arithmetic, logical, control and input/output (I/O) operations specified by the instructions."

CPU Components

The CPU is divided in two main components:

- **Arithmetic-Logic Unit**: Calculates arithmetic and logic operation between two numbers.
- **Control Unit**: Directs the operations of the processor.

Memory

Definition (Memory)

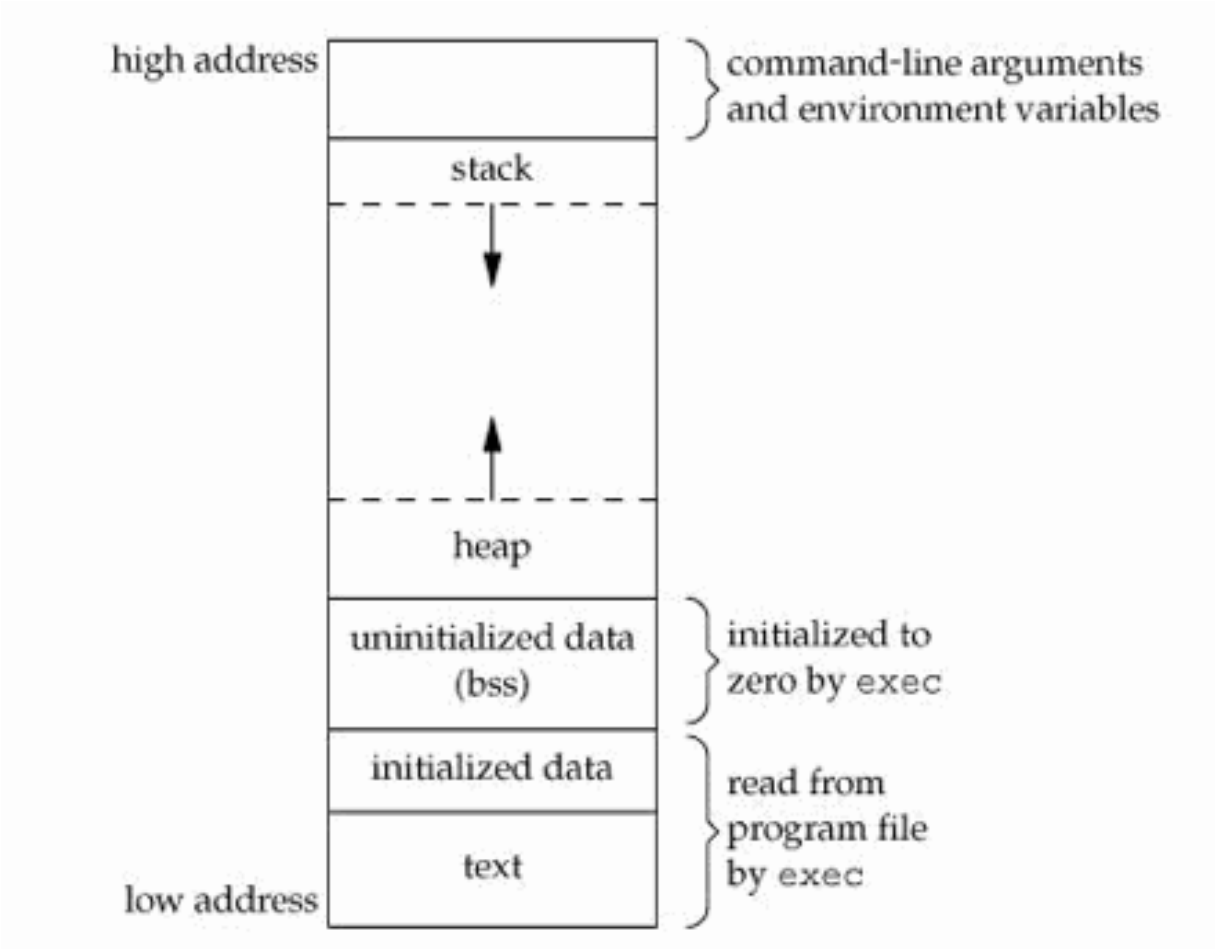
"Devices used to store information for immediate use in a computer. This information is usually data or instructions."

Memory Description

The memory is featured by containing the running programs which are usually divided in:

- **Instructions**: an order to the processor.
- **Data**: information which is managed.

Memory Example



Memory Protections

Memory protections control memory access rights.

It aims to prevent erroneous or malicious accesses to specific memory sections.

It also separates program spaces.

Memory Protections: Segmentation

Divides the memory into segments.

The memory is referenced in terms of segments and offsets.

The Global and Local Descriptor Tables solve the references to the physical memory.

Memory Protections: Paging

Divides the memory into equal size blocks (pages).

Virtual Memory Hardware allows page isolation and fragmentation on physical memory.

Pages can be flagged as protected.

I/O Peripheral Devices

Definition (I/O Devices)

"The set of devices used to communicate between a computer and the outside world or another computer."

I/O Peripheral Devices: Examples

Some examples of I/O devices:

- Printer.
- Screen.
- Mouse.
- Keyboard.
- USB stick.

Programs & Instructions (I)

Definition (Program)

A program is a sequence of instructions which perform a specific task.

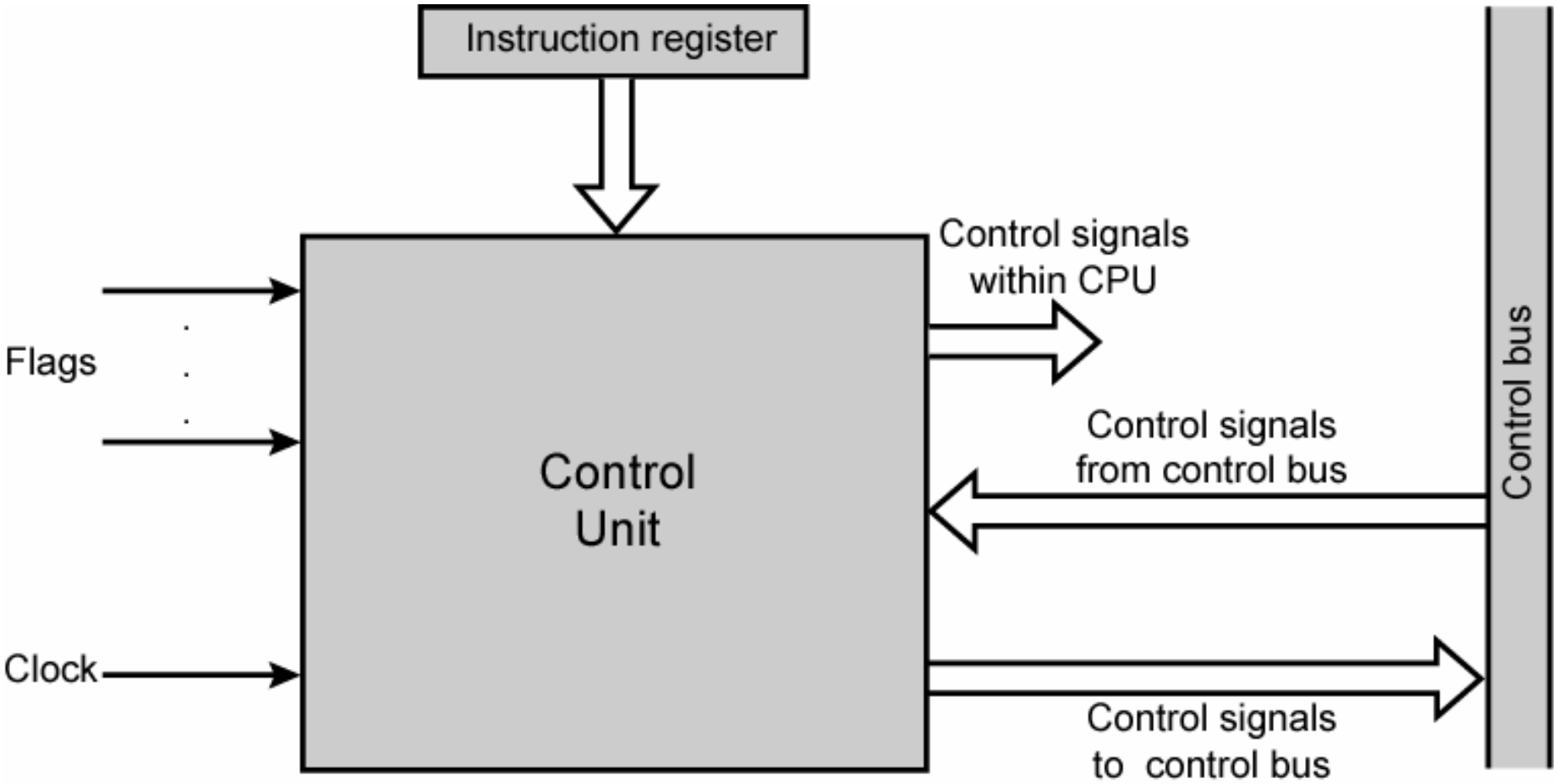
Definition (Instruction)

An instruction is an order to the processor.

Programs & Instructions (II)

The program is read by the machine. Its execution is divided in the following steps:

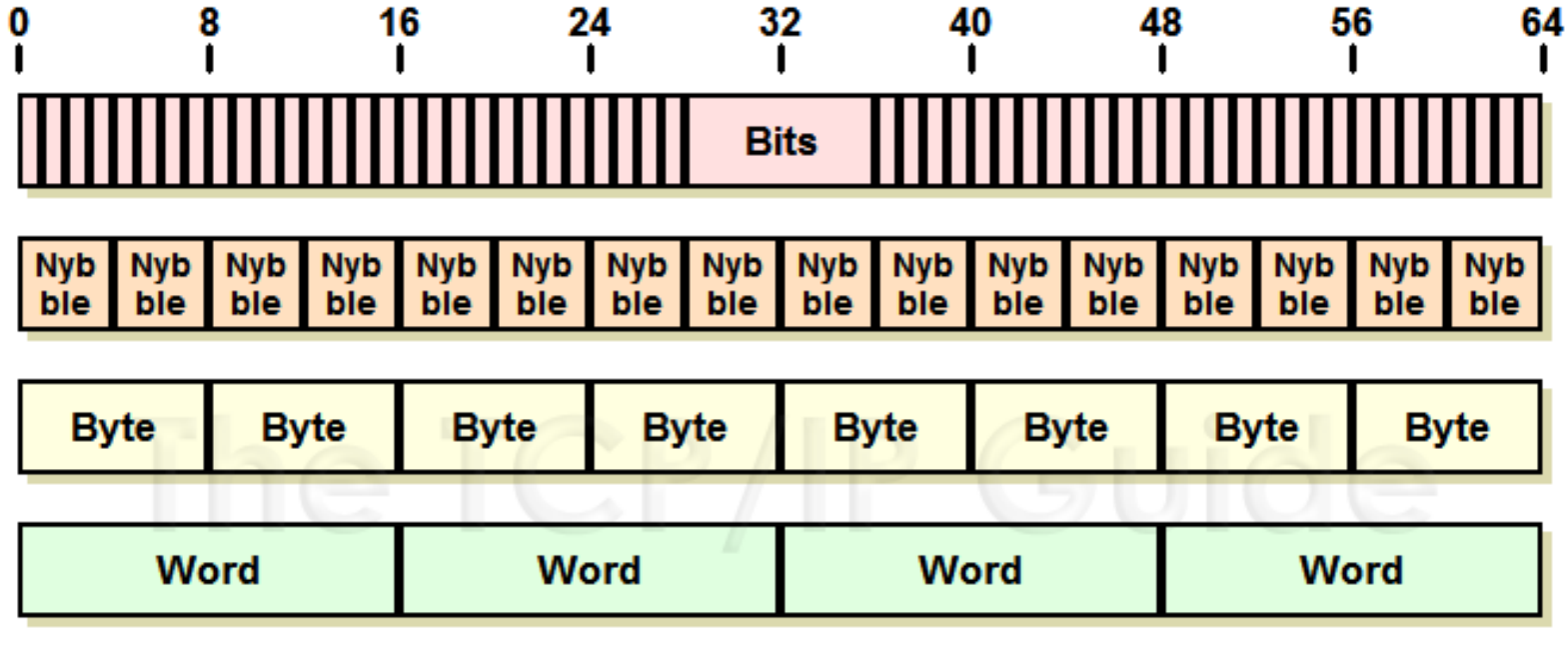
1. The program is introduced in the main memory.
2. The Operating System asks the computer to move the control to the i^{th}-position of the main memory (where the program stars).
3. The Control Unit repeats the following steps: it takes the program instructions **(fetching)**, decode them **(decoding)**, executes the instruction **(managing execution)** and then stores the results **(storing)**.


Instruction register
Control signals within CPU
Flags
Control Unit
Control signals from control bus
Control bus
Clock
Control signals to control bus

Instructions in Memory

The memory stores the instructions and data. This information is usually organized as:

- **Bits**: the smallest unit.
- **Bytes** (8 bits): smallest location.
- **Words** (16/32/64 bits): unit of data for the processor.



Bases

Decimal (Base 10)	Binary (Base 2)	Hex (Base 16)
00	0000b	0x00
01	0001b	0x01
02	0010b	0x02
03	0011b	0x03
04	0100b	0x04
05	0101b	0x05
06	0110b	0x06
07	0111b	0x07
08	1000b	0x08
09	1001b	0x09
10	1010b	0x0A
11	1011b	0x0B
12	1100b	0x0C
13	1101b	0x0D
14	1110b	0x0E
15	1111b	0x0F

Negative Numbers

"One's complement" = flip all bits. 0 to 1, 1 to 0.

"Two's complement" = one's complement + 1.

Negative numbers are defined as the "two's complement" of the positive number.

Number	One's Comp.	Two's Comp.
00000001b : 0x01	11111110b: 0xFE	11111111b: 0xFF (-1)
00000100b : 0x04	11111011b: 0xFB	11111100b: 0xFC (-4)
00011010b : 0x1A	11100101b: 0xE5	11100110b: 0xE6 (-26)
01010000b : 0x50	10101111b: 0xAF	10110000b: 0xB0 (-80)

Instruction Structure

The instructions are usually composed by:

- **Operand Codes (Opcodes)**: specifies the operation to perform.
- **Memory Address**: specifies the memory section affected by this operation. It can be used for reading or writing and it usually contains variables.

Example:

Opcode	Addr 1	Addr 2

Instructions Set

Definition (Instructions Set)

Collection of instructions that can be executed by the processor.

There are two main instruction set:

- **CISC**: Complex Instruction Set Computer.
- **RISC**: Reduced Instruction Set Computer.

CISC vs. RISC

Features of CISC:

- A lot of special purpose instructions.
- Variable-length instructions, between 1 and 16 bytes long.
- Examples: Intel.

Features of RISC:

- Typically more registers, less and fixed-size instructions.
- Examples: PowerPC, ARM, SPARC, MIPS

Endianness (I)

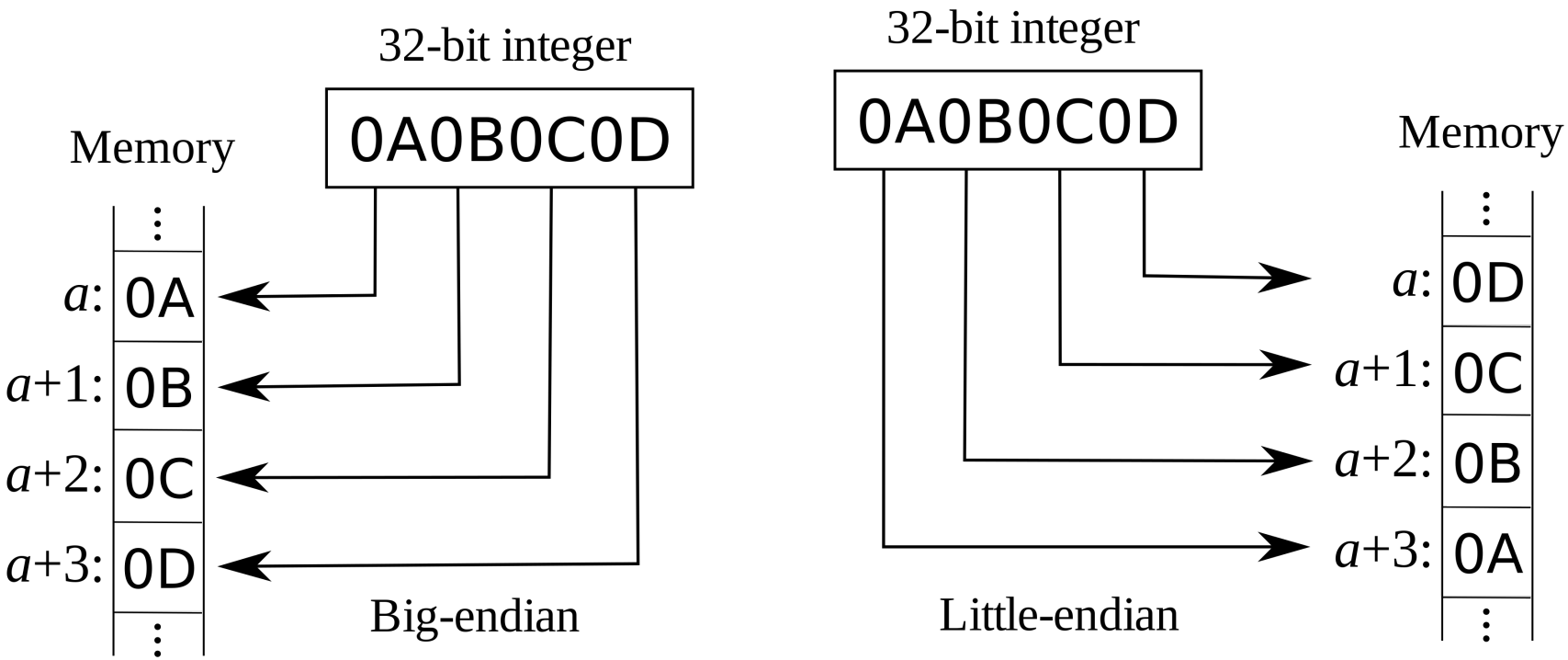
Definition (Endianness)

"The ordering or sequencing of bytes of a word of digital data in computer memory storage or during transmission."

Types of Endian:

- **Little Endian**: The least significant byte of a word or larger is stored in the lowest address.
- **Big Endian**: The least significant byte of a word or larger is stored in the highest address.

Endianness (II)



Endianness (III)

Examples:

- Intel is Little Endian.
- Network traffic is Big Endian.

Registers (I)

Definition (Registers)

"Registers are small memory storage areas built into the processor (still volatile memory)"

8 registers + the instruction pointer which points at the next instruction to execute.

On x86-32, registers are 32 bits long.

On x86-64, they are 64 bits

Registers (II)

Frequent but not mandatory use:.

EAX: Stores function return values.

EBX: Base pointer to the data section.

ECX: Counter for string and loop operations.

EDX: I/O pointer.

Registers (III)

ESI: Source pointer for string operations.

EDI: Destination pointer for string operations.

ESP: Stack pointer.

EBP: Stack frame base pointer.

EIP: Pointer to next instruction to execute (“instruction pointer”).

Flags

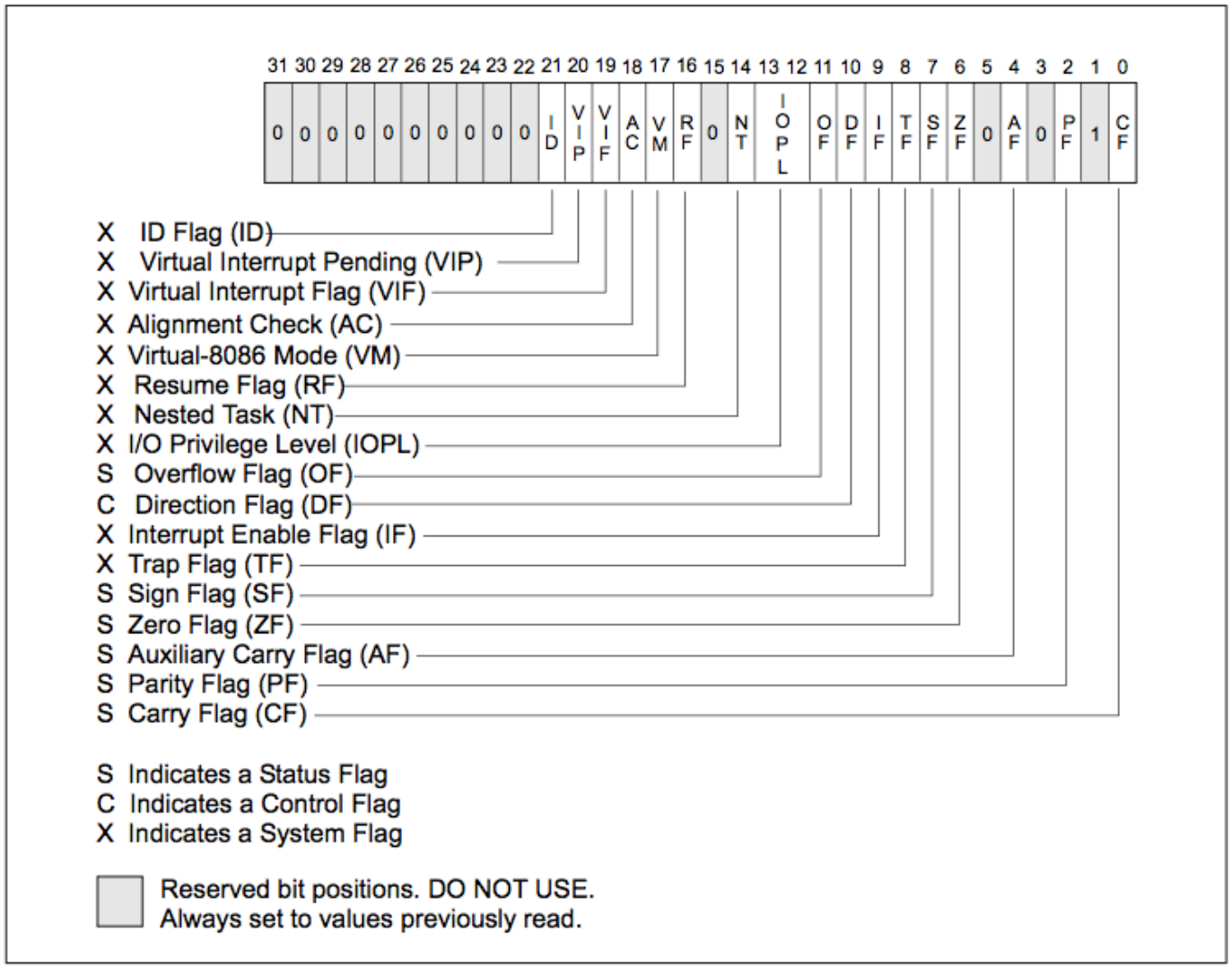
Definition

"A Flag is a bit of a register that is activated when a specific event happens."

EFLAGS register holds many single bit flags.

CF		PF		AF		ZF	SF	TF	IF	DF	OF	IOPL	NT	
RF	VM	AC	VIF	VIP	ID									

Flags


31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
0 0 0 0 0 0 0 0 0 0 ID VIP VIF AC VM RF 0 NT IOPL OF DF IF TF SF ZF 0 AF 0 PF 1 CF
X ID Flag (ID)
X Virtual Interrupt Pending (VIP)
X Virtual Interrupt Flag (VIF)
X Alignment Check (AC)
X Virtual-8086 Mode (VM)
X Resume Flag (RF)
X Nested Task (NT)
X I/O Privilege Level (IOPL)
S Overflow Flag (OF)
C Direction Flag (DF)
X Interrupt Enable Flag (IF)
X Trap Flag (TF)
S Sign Flag (SF)
S Zero Flag (ZF)
S Auxiliary Carry Flag (AF)
S Parity Flag (PF)
S Carry Flag (CF)
S Indicates a Status Flag
C Indicates a Control Flag
X Indicates a System Flag
Reserved bit positions. DO NOT USE.
Always set to values previously read.

Stack (I)

Definition (Stack)

"The stack is a conceptual area of main memory (RAM) which is designated by the OS when a program is started."

Different OS start it at different addresses by convention.

A stack is a Last-In-First-Out (LIFO/FILO) data structure where data is "pushed" on to the top of the stack and "popped" off the top.

By convention the stack grows toward lower memory addresses. Adding something to the stack means the top of the stack is now at a lower memory address.

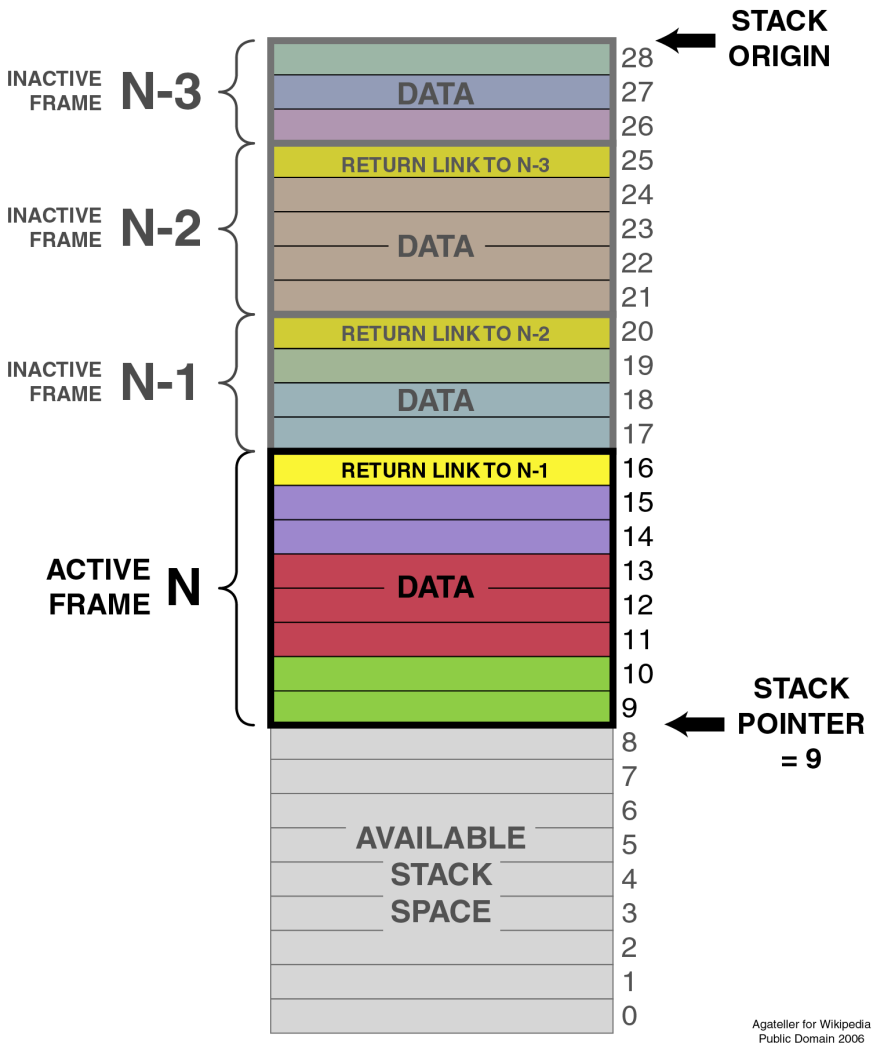
Stack (II)

ESP points to the top of the stack, the lowest address which is being used.

The stack keeps track of which functions were called before the current one, it holds local variables and is frequently used to pass arguments to the next function to be called.

A firm understanding of what is happening on the stack is **essential** to understand program's operation.

Stack (III)


INACTIVE FRAME N-3
INACTIVE FRAME N-2
INACTIVE FRAME N-1
ACTIVE FRAME N
DATA
RETURN LINK TO N-3
DATA
RETURN LINK TO N-2
DATA
RETURN LINK TO N-1
DATA
AVAILABLE STACK SPACE
STACK ORIGIN
STACK POINTER = 9
28
27
26
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0
Agateller for Wikipedia
Public Domain 2006

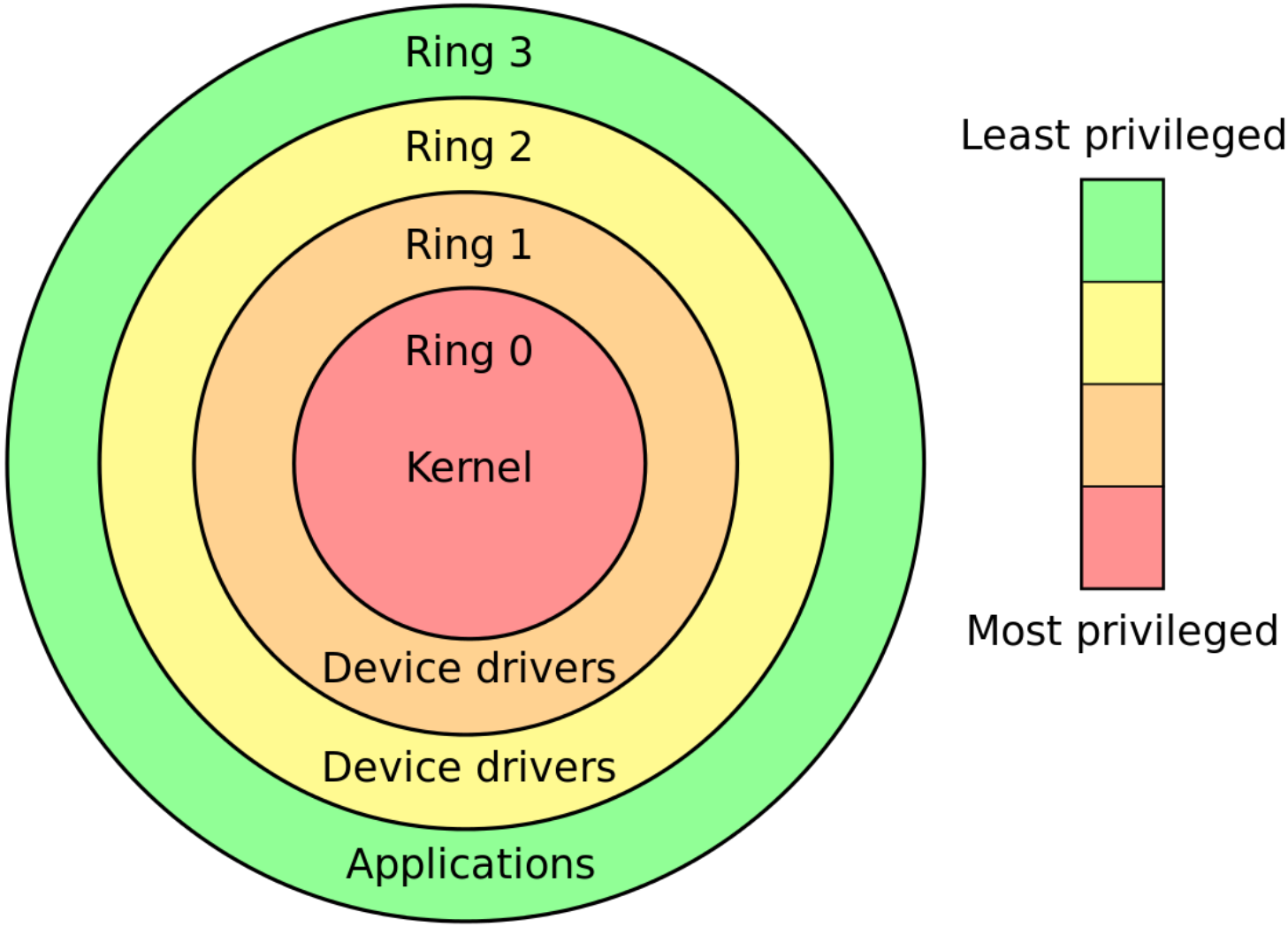
Protection Rings

To ensure some extra protection levels the CPU has different modes, hierarchically separated.

Some operative systems only use two modes for compatibility reasons.

The higher level uses gates to access low level's resources such as I/O devices.

Protection Rings



Protection Rings: modes

The two main modes of current OS are supervision mode and user mode.

The OS runs in supervision mode and has total control of memory, threads, flags, etc. Programs running in this mode must not fail.

The normal applications run in user mode and they have limited privileges.

Questions?