**CEN 214 Microprocessors Lab**
**Assignment 7**

## New Instructions

### INT imm8 (0..255)

The INT imm8 instruction generates a call to the interrupt or exception handler specified with the destination operand (see the section titled "Interrupts and Exceptions" in Chapter 6 of the Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 1). The destination operand specifies a vector from 0 to 255, encoded as an 8-bit unsigned intermediate value. Each vector provides an index to a gate descriptor in the IDT. The first 32 vectors are reserved by Intel for system use. Some of these vectors are used for internally generated exceptions. The INT imm8 instruction is the general mnemonic for executing a software-generated call to an interrupt handler.

***INT 10h / AH = 0Eh*** *- teletype output.*

*input:*
***AL*** *= character to write.*

*this functions displays a character on the screen, advancing the cursor and scrolling the screen as necessary. the printing is always done to current active page.*

*example:*

```
mov al, 'a'
mov ah, 0eh          ; note: on specific systems this
int 10h              ; function may not be supported in graphics mode.
```

### LEA reg16, variable

Computes the effective address of the second operand (the source operand) and stores it in the first operand (destination operand). The source operand is a memory address (offset part) specified with one of the processors addressing modes; the destination operand is a general-purpose register. The address-size and operand-size attributes affect the action performed by this instruction, as shown in the following table. The operand-size attribute of the instruction is determined by the chosen register; the address-size attribute is determined by the attribute of the code segment.

```
character  db  231    ; Character ç
lea bx, character     ; loads variable character's address to BX
mov al, [bx]
mov ah, 0eh           ; note: on specific systems this
int 10h               ; function may not be supported in graphics mode.
```

**Examples**

**1.** Write a program that writes your name to the screen. You must use a DB variable. (e.g. *name DB "John DOE"* )

**2.** Write a program that writes your name to the screen backwards. You must use a DB variable. (e.g. *name DB "John DOE"* )

**3.** Write a program that writes a given lowercase string variable to the screen with uppercase characters. (e. g. *name DB "John DOE" --> JOHN DOE)*