Mansoura University
Faculty of Computers and Information
Department of Computer Science
First Semester: 2020-2021

[CS214P] Assembly Language

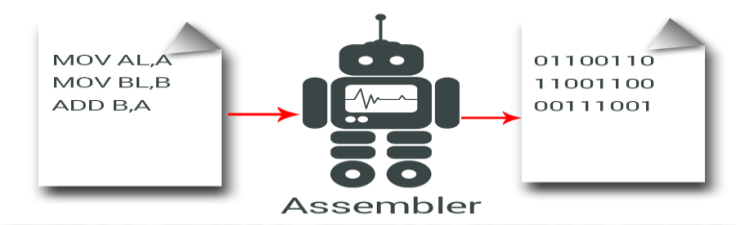
Grade: Third Year (Computer Science)

Sara El-Metwally, Ph.D.

Faculty of Computers and Information,

Mansoura University,

Egypt.



Computer Science Department
Faculty of Computers and Information
Mansoura University

Assembly Language

"Examining Computer Memory and Executing Instructions"

Sara El-Metwally, Ph.D.
Faculty of Computers and Information,
Mansoura University, Egypt.

Email: sarah_almetwally4@mans.edu.eg
sara.elmetwally.2007@gmail.com

Using INT Instruction

(INT)

- INT: is an assembly language instruction for x86 processors that generates a software interrupt.

- When written in assembly language, the instruction is written like this: INT X
  - where X is the software interrupt that should be generated (How many?).

Using INT Instruction

(INT)

- There are different types of INT operations, some of which require a function code in the AH register to request a specific action.

- We will use P command to execute through the whole interrupt routine.

Using INT Instruction

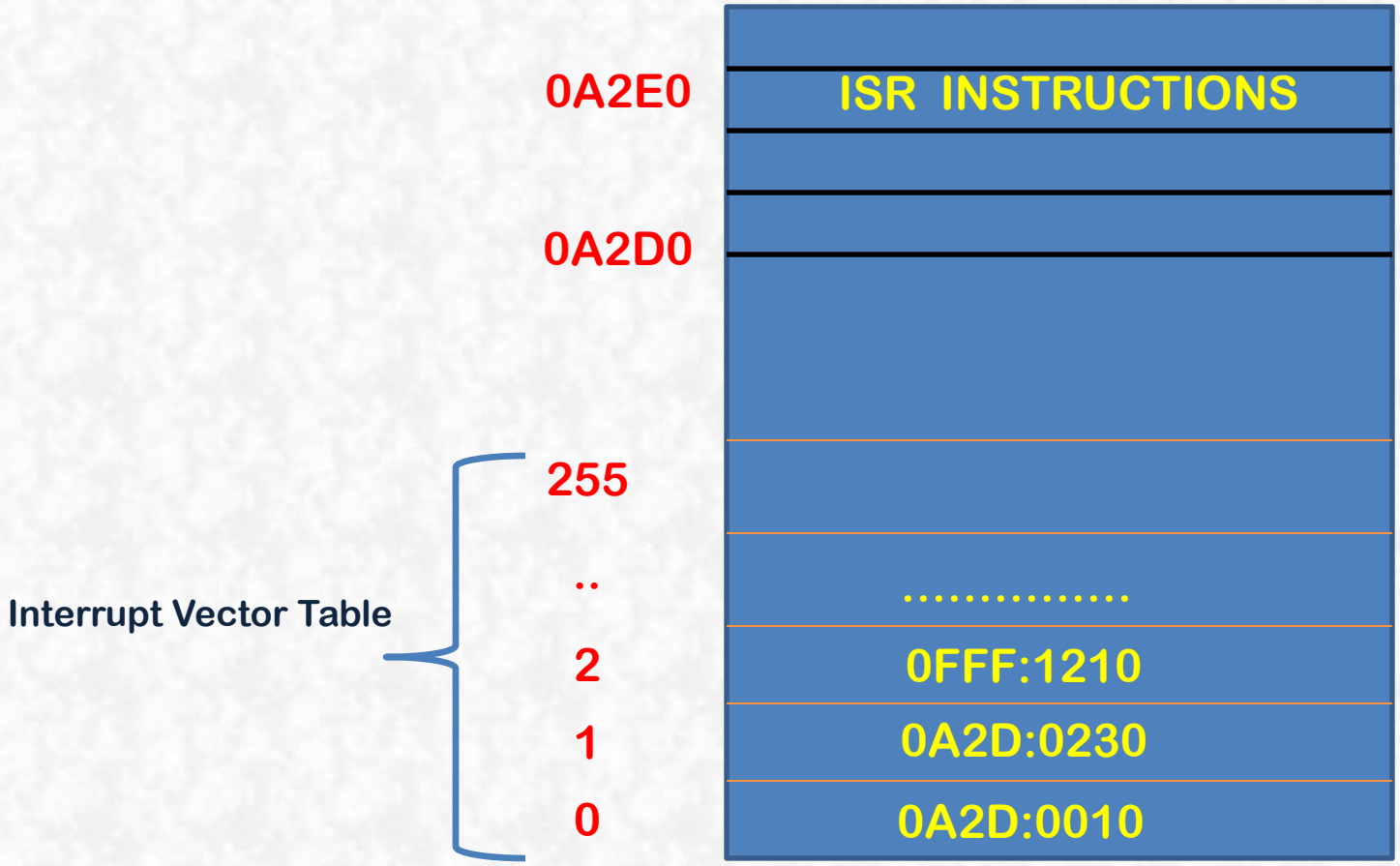
(INT)

- There are different types of INT operations, some of which require a function code in the AH register to request a specific action.

- We will use P command to execute through the whole interrupt routine.

Using INT Instruction

(INT)



Using INT Instruction

(current date, INT 21, AH=2A)

```
Y:\>DEBUG.EXE
-A 100
073F:0100 MOV AH,2A
073F:0102 INT 21
073F:0104 JMP 100
073F:0106
-T

AX=2A00  BX=0000  CX=0000  D
DS=073F  ES=073F  SS=073F  C
073F:0102 CD21          INT
-P

AX=2A06  BX=0000  CX=07E4  DX=0B07  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0104   NV UP EI PL NZ NA PO NC
073F:0104 EBFA          JMP     0100
```

Enter hex number:

07E4 16

Convert | Reset | Swap

Decimal number:

2020 10

Day of the week, 0 =Sunday

DH: month (01 H to 0CH)

DL: Day (01 H to 1FH)

Using INT Instruction

(current time, INT 21, AH=2C)

```
-A 100
073F:0100 MOV AH,2C
073F:0102 INT 21
073F:0104 JMP 100
073F:0106
-R
AX=0000  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0100   NV UP EI PL NZ NA PO NC
073F:0100 B42C          MOV     AH,2C
-T

AX=2C00  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0102   NV UP EI PL NZ NA PO NC
073F:0102 CD21          INT     21
-P

AX=2C00  BX=0000  CX=0F36  DX=1F29  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0104   NV UP EI PL NZ NA PO NC
073F:0104 EBFA          JMP     0100
```

seconds

minutes

Hours in 24-hour format

Using INT Instruction

(Displaying, INT 21, AH=09, starting address DX)

```
-A 100
073F:0100 MOV AH,09
073F:0102 MOV DX, 109
073F:0105 INT 21
073F:0107 JMP 100
073F:0109 DB 'SARA ELMETWALLY','$'
073F:0119
-
```

Using INT Instruction

(Displaying, INT 21, AH=09, starting address DX)

```
-R
AX=0000  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0100   NV UP EI PL NZ NA PO NC
073F:0100 B409          MOV     AH,09
-T

AX=0900  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0102   NV UP EI PL NZ NA PO NC
073F:0102 BA0901        MOV     DX,0109
-T

AX=0900  BX=0000  CX=0000  DX=0109  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0105   NV UP EI PL NZ NA PO NC
073F:0105 CD21          INT     21
-P
SARA ELMETWALLY
AX=0900  BX=0000  CX=0000  DX=0109  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0107   NV UP EI PL NZ NA PO NC
073F:0107 EBF7          JMP     0100
```

Using INT Instruction

(Accept chars from keyboard)

(INT 16, AH=10, AL=result)

```
-A 100
073F:0100 MOV AH,10
073F:0102 INT 16
073F:0104 JMP 100
073F:0106
-T

AX=1000  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0102   NV UP EI PL NZ NA PO NC
073F:0102                 INT     16
-P

AX=1E41                   DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F                   CS=073F  IP=0104   NV UP EI PL NZ NA PO NC
073F:0104                 JMP     0100
-
```

Hex	Character
40H	@
41H	A
42H	B
43H	C
44H	D
45H	E

Using “ PTR” Operator

```
100 MOV AX, [11A]
103 ADD AX, [11C]
107 ADD AX, 25
10A MOV[11E],AX
10D MOV WORD PTR [120], 25
113 MOV BYTE PTR [122], 30
118 JMP 100
11A DB 14 23
11C DB 05 00
11E DB 00 00
120 DB 00 00 00
```

Using “ PTR” Operator

```
Y:\>DEBUG.EXE
-A 100
073F:0100 MOV AX, [11A]
073F:0103 ADD AX, [11C]
073F:0107 ADD AX,25
073F:010A MOV [11E],AX
073F:010D MOV WORD PTR [120],25
073F:0113 MOV BYTE PTR [122],30
073F:0118 JMP 100
073F:011A DB 14 23
073F:011C DB 05 00
073F:011E DB 00 00
073F:0120 DB 00 00 00
073F:0123
-
```

```
AX=2314
```

```
AX=2319
```

```
ADD     AX,0025
```

```
AX=233E
```

```
-D 073F:0120
073F:0120  25 00 00 00 00 00
073F:0130  00 00 00 00 00 00
```

```
-D 073F:0122
073F:0120        30 00 00 00 00
073F:0130  00 00 00 00 00 00 00
```

```
-D 073F:011E
073F:0110                                          3E 23
073F:0120  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
```

Using “ PTR” Operator

```
-R
AX=0000  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0100   NV UP EI PL NZ NA PO NC
073F:0100 A11A01        MOV     AX,[011A]                        DS:011A=2314
-T

AX=2314  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0103   NV UP EI PL NZ NA PO NC
073F:0103 03061C01      ADD     AX,[011C]                        DS:011C=0005
-T

AX=2319  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0107   NV UP EI PL NZ NA PO NC
073F:0107 052500        ADD     AX,0025
-T
```

Assembler IDE

Secure | https://sourceforge.net/projects/guitasm8086/

sourceforge | Search | Browse | Enterprise | Blog | Articles

SOLUTION CENTERS | Resources | Newsletters | Cloud Storage Providers | Business VoIP Providers | Internet S

Ad covered content | Seen this ad multiple times | Ad was inappropriate | Not interested in this ad

Advertisement - Report

Home / Browse / Development / Assemblers / GUI Turbo Assembler (TASM)

GUI Turbo Assembler (TASM)

A 32-64bit Multilingual IDE for Assembly Language with TASM & TLINK

Brought to you by: ljnath

Summary | Files | Reviews | Support | Discussion | Tickets

★ 5.0 Stars (3)

1,066 Downloads (This Week)

Last Update: 5 days ago

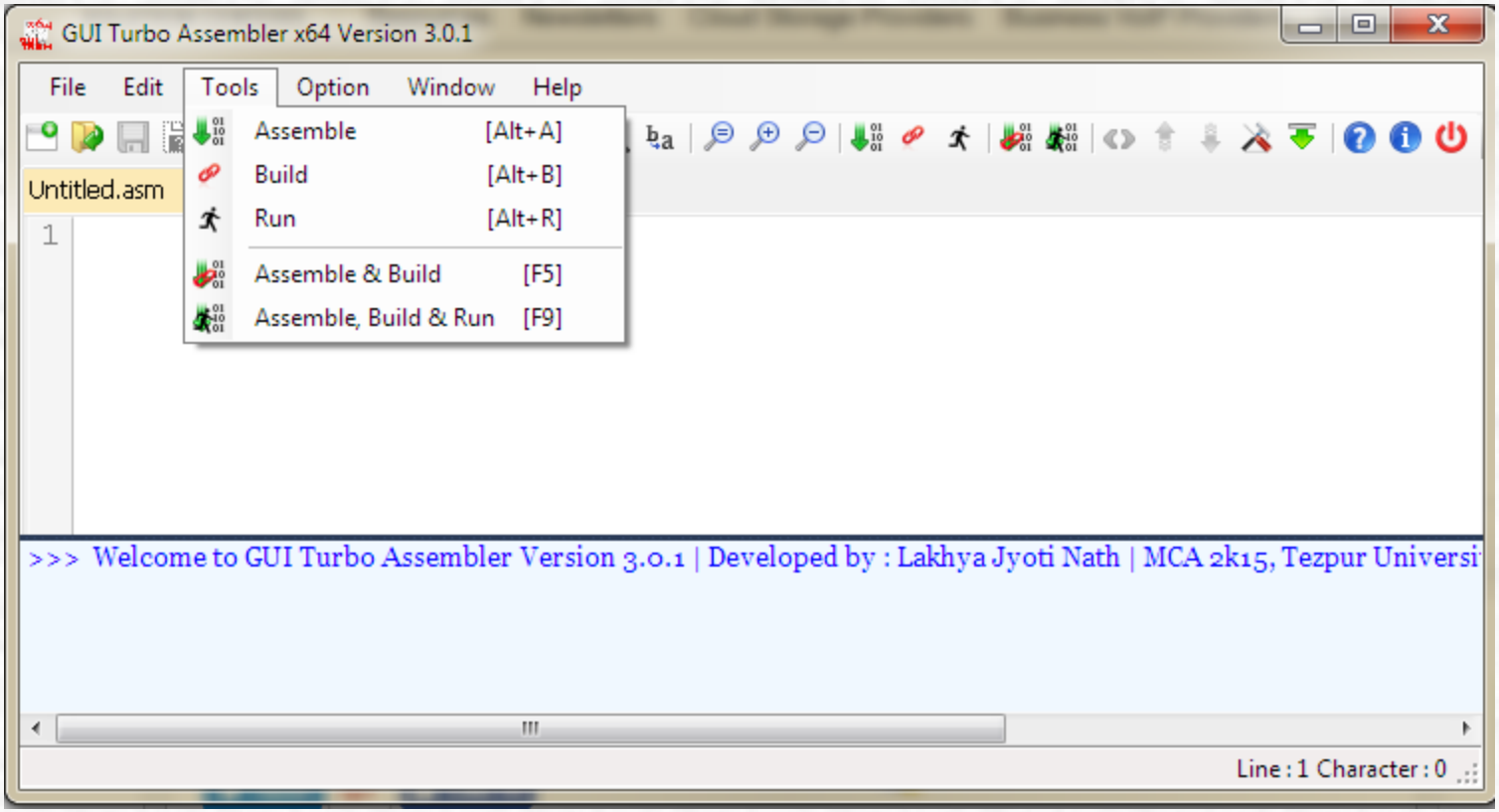
Tweet | G+ | Like 80

Download

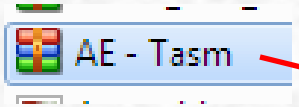
GUI Turbo Assembler v3.0.1.msi

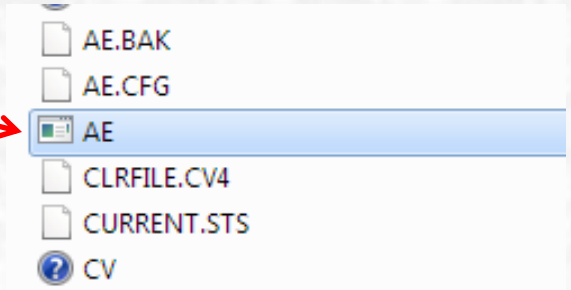
Browse All Files

Assembler IDE



Assembler IDE






```
Z:\>mount y: D:\College-Courses\Assembly\AE\Tasm
Drive Y is mounted as local directory D:\College-Courses\Assembly\AE\Tasm\

Z:\>y:

Y:\>AE.EXE_
```

Assembler IDE