

Philadelphia University Faculty of Engineering

Marking Scheme

Examination Paper Department of CE

Module: Microprocessors (630313)

First Exam

Second Semester

Date: 02/04/2019

Section 1

Weighting 20% of the module total

Lecturer:

Coordinator:

Internal Examiner:

Dr. Qadri Hamarsheh Dr. Qadri Hamarsheh Dr. Naser Halasa

Marking Scheme Microprocessors (630313)

The presented exam questions are organized to overcome course material, the exam contains 5 questions; *all questions* are compulsory requested to be answered. Thus, the student is permitted to answer any question out of the existing ones in this section.

Marking Assignments

The following scheme shows the marks assignments for each question. They show also the steps for which a student can get marks along the related procedure he/she achieves.

Question 1 This question is attributed with 5 marks, if answered properly, The answer for this question as the following:

1) During which phase of the instruction execution cycle is the program counter (**IP register**) incremented?

| | a) | decode | b) |) | execute | |
|----|------------------|-----------------------|-------------------------|-------|------------------------|----|
| | c) | operand fetch | u d) | | fetch | |
| 2) | The size of | each segment in 8 | 086 is | _ | | |
| | a) | 16kb | b) | | 24 kb | |
| | c) | 50 kb | d) | | 64 kb | |
| 3) | The Overf | low flag is based | on signed arithmetic | • | | |
| | a) | True | b) | | False | |
| 4) | Which of th | nese values is the n | nost acceptable for the | e da | ta type REAL4 ? | |
| | a) | 1275 | b) | | 'Q' | |
| | c) | 23.26 | d) | | "Jennifer",0 | |
| 5) | One of the | following memo | ry models combine | s the | e data and code parts | ;: |
| | a) Sn | nall | | b) | Tiny | |
| | c) M | eduim | | d) | Huge | |

Question 2 This question is attributed with 3 marks if answered properly, The answer for this question as the following:

| | Solution |
|--------------|----------------------------------------------------------------------------------------------|
| enera | te 16-bit MS-DOS Programs |
| \checkmark | Advantages |
| | enables calling of MS-DOS and BIOS functions |
| | no memory access restrictions |
| ✓ | Disadvantages |
| | must be aware of both segments and offsets |
| | cannot call Win32 functions (Windows 95 onward) |
| | limited to 640K program memory |
| \checkmark | Requirements |
| | INCLUDE Irvine16.inc |
| | • Two additional instructions are inserted at the beginning of the startup procedure (main) |
| | Initialize DS to the data segment using predefined MASM constant @data:: |
| | mov ax,@data |
| | mov ds,ax |

<u>Question 3</u> This question is attributed with 5 marks, if answered properly. The answer for this question as the following: **a**)

| Ñ | Instruction | Answer |
|----|------------------|--------|
| 1) | mov word1, byte2 | I |
| 2) | mov word2,10000h | I |
| 3) | mov si, ds | L |
| 4) | movzx ax, bytel | L |
| 5) | movsx dl,al | I |

b)

| mov dx,word3 | DX = 7FFFh |
|---------------------------------------|-------------------|
| movsx eax,bytel | EAX = FFFFFFFFh |
| mov dh,al | DH = FFh |
| mov ax,[word3+2] | AX = 8000h |
| mov eax,[word3+4] | EAX = I (illegal) |
| · · · · · · · · · · · · · · · · · · · | |

Question 4 This question is attributed with 3 marks, if answered properly. The answer for this question as the following:

| Original Program | Your answer |
|--------------------------------------|--------------------------------------|
| INCLUDE irvine32.inc | INCLUDE irvine32.inc |
| .data | .data |
| arrayA word 4Ah, 3Ch, 2h, 5 DUP (1h) | arrayA word 4Ah, 3Ch, 2h, 5 DUP (1h) |
| varl word ffffh | varl word <mark>O</mark> ffffh |
| .code | .code |
| main PROC | main PROC |
| mov cx,4Ch | mov cx,4Ch |
| xchg cx, arrayA+4 | xchg cx, arrayA+4 |
| mov ax, arrayA+5 | mov ax, arrayA+5 |
| mov bl, varl | mov <mark>bx</mark> , varl |
| add ax, bx | add ax, bx |
| call dumpregs | call dumpregs |
| exit | exit |
| main ENDP | main ENDP |
| | end main |

Question 5 This question is attributed with 4 marks, if answered properly. The complete code for this question as the following:

| Title Simple Arithmetic .data J sword K sword L sword U1 word U2 word U3 word .code (1 mark) .code (1 mov K, -2 mov U1, 254 (1 mark) mov ax, J add ax, K mov L, ax mov ax, u1 add ax, u2 mov u3, ax main PROC main ENDP | | | | Solution |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|--------------|-----|-----------|
| .data J sword ? K sword ? L sword ? U1 word ? U2 word ? U3 word ? .code (1 mark) .code mov U1, 254 mov U2, 22 (1 mark) mov ax, J add ax, K mov L, ax mov ax, u1 add ax, u2 mov u3, ax main PROC main ENDP | Title Simp | ole Arithmet | tic | |
| J sword ? K sword ? L sword ? U1 word ? U2 word ? U3 word ? U3 word ? .code (1 mark) .code mov J, 3 mov U1, 254 (1 mark) mov ax, J add ax, K mov L, ax mov ax, U1 add ax, u2 mov u3, ax main PROC main ENDP | .data | | | |
| K sword ? L sword ? U1 word ? U2 word ? U3 word ? .code mov J, 3 mov K, -2 mov U1, 254 mov U2, 22 (1 mark) mov ax, J add ax, K mov L, ax mov ax, u1 add ax, u2 mov u3, ax main PROC main ENDP | J | sword | ? | |
| L sword ? U1 word ? U2 word ? U3 word ? .code mov J, 3 mov K, -2 mov U1, 254 mov 22, 22 (<i>l mark</i>) mov ax, J add ax, K mov L, ax mov ax, u1 add ax, u2 mov u3, ax main PROC main ENDP | K | sword | ? | |
| U1 word ? U2 word ? U3 word ? (1 mark) .code mov J, 3 mov K, -2 mov U1, 254 mov U2, 22 mov ax, J add ax, K mov L, ax mov ax, u1 add ax, u2 mov u3, ax main PROC main ENDP | L | sword | ? | |
| U2 word ? (1 mark) .code mov J, 3 (1 mark) mov K, -2 mov U1, 254 (1 mark) mov U2, 22 (1 mark) mov ax, J add ax, K mov L, ax mov ax, u1 add ax, u2 mov u3, ax main PROC main ENDP | U1 | word | ? | |
| U3 word ? (1 mark) .code mov J, 3 mov K, -2 mov U1, 254 mov U2, 22 (1 mark) mov ax, J add ax, K mov L, ax mov ax, u1 add ax, u2 mov u3, ax main PROC main ENDP | U2 | word | ? | |
| .code mov J, 3 mov K, -2 mov U1, 254 mov U2, 22 (1 mark) mov ax, J add ax, K mov L, ax mov ax, u1 add ax, u2 mov u3, ax main PROC main ENDP | U 3 | word | ? | (1 mark) |
| mov J, 3 mov K, -2 mov U1, 254 mov U2, 22 (1 mark) mov ax, J add ax, K mov L, ax mov ax, u1 add ax, u2 mov u3, ax main PROC main ENDP | .code | | | |
| mov K, -2 mov U1, 254 mov U2, 22 (1 mark) mov ax, J add ax, K mov L, ax mov ax, u1 add ax, u2 mov u3, ax main PROC main ENDP | mov | J, 3 | | |
| mov U1, 254 mov U2, 22 (1 mark) mov ax, J add ax, K mov L, ax mov ax, u1 add ax, u2 mov u3, ax main PROC main ENDP | mov | К, -2 | | |
| mov U2, 22 (1 mark) mov ax, J add ax, K mov L, ax mov ax, u1 add ax, u2 mov u3, ax main PROC main ENDP | mov U1, 254 | | | |
| mov ax, J add ax, K mov L, ax mov ax, ul add ax, u2 mov u3, ax main PROC main ENDP | mov U2, 22 | | | (1 mark) |
| add ax, K mov L, ax mov ax, ul add ax, u2 mov u3, ax main PROC main ENDP | mov ax, J | | | |
| mov L, ax mov ax, ul add ax, u2 mov u3, ax main PROC main ENDP | add ax, K | | | |
| mov ax, ul add ax, u2 mov u3, ax main PROC main ENDP | mov L, ax | | | |
| add ax, u2 mov u3, ax main PROC main ENDP | mov ax, ul | | | |
| mov u3, ax main PROC main ENDP | add ax, u2 | | | |
| main PROC main ENDP | mov | u3, ax | | |
| main ENDP | main PRC | DC | | |
| | main ENI |)P | | |
| END main (2 marks) | END mai | n | | (2 marks) |