EECE.3170: Microprocessor Systems Design I

Spring 2016

Exam 1
February 19, 2016

Name: _____

Section (circle 1): **201** *(MWF 9-9:50)* **202** *(MWF 10-10:50)*

For this exam, you may use a calculator and one 8.5" x 11" double-sided page of notes. All other electronic devices (e.g., cellular phones, laptops, PDAs) are prohibited. If you have a cellular phone, please turn it off prior to the start of the exam to avoid distracting other students.

The exam contains 5 questions. The first four questions will give you a total of 100 points; the fifth question is an extra credit problem worth 10 points. **In order to receive any extra credit for Question 5, you must clearly demonstrate that you have made a significant effort to solve each of the first four questions.**

Please answer the questions in the spaces provided. If you need additional space, use the back of the page on which the question is written and clearly indicate that you have done so.

You will be provided with two pages (1 double-sided sheet) of reference material for the exam: a list of the x86 instructions we have covered thus far. You do not have to submit this sheet when you turn in your exam.

You will have 50 minutes to complete this exam.

Q1: Multiple choice	/ 20
Q2: Data transfers and memory addressing	/ 30
Q3: Arithmetic instructions	/ 30
Q4: Logical instructions	/ 20
TOTAL SCORE	/ 100
Q5: EXTRA CREDIT	/ 10

1. (20 points, 5 points per part) ***Multiple choice***
For each of the multiple choice questions below, clearly indicate your response by circling or underlining the single choice you think best answers the question.

a. Given `AL = C3h, BL = 1Bh,` and `CF = 1`, what is the final result of the instruction `ADC AL, BL`?

i. `AL = C3h, CF = 0`

ii. `AL = DEh, CF = 0`

iii. `AL = DEh, CF = 1`

iv. `AL = DFh, CF = 0`

v. `AL = DFh, CF = 1`

b. Given `AL = 96h` and `CF = 0`, what is the final result of the instruction `RCR AL, 3`?

i. `AL = 12h, CF = 1`

ii. `AL = 92h, CF = 1`

iii. `AL = B2h, CF = 0`

iv. `AL = D2h, CF = 1`

v. `AL = F2h, CF = 1`

1 (continued)

c. If `AX = 3A4Bh`, which of the following instructions will set `CF = 1`?

A. `BT  AX, 3`
B. `BTR  AX, 12`
C. `BTS  AX, 1`
D. `BTC  AX, 6`

i. Only A

ii. Only B

iii. A and D

iv. B and C

v. All of the above (A, B, C, D)

d. If `AX = 10F0H`, which of the following choices correctly shows the results of performing the two bit scan instructions (`BSF` and `BSR`) on this register?

i.
```
BSF DX, AX      → ZF = 1, DX = 0004h
BSR DX, AX      → ZF = 1, DX = 000Ch
```

ii.
```
BSF DX, AX      → ZF = 1, DX = 0004h
BSR DX, AX      → ZF = 1, DX = 0012h
```

iii.
```
BSF DX, AX      → ZF = 0, DX = 0004h
BSR DX, AX      → ZF = 0, DX = 000Ch
```

iv.
```
BSF DX, AX      → ZF = 1, DX = 0003h
BSR DX, AX      → ZF = 1, DX = 0005h
```

v.
```
BSF DX, AX      → ZF = 0, DX unchanged
BSR DX, AX      → ZF = 0, DX unchanged
```

2. (30 points) ***Data transfers and memory addressing***

For each data transfer instruction in the sequence shown below, list all changed registers and/or memory locations and their final values. If memory is changed, be sure to explicitly list **all changed bytes**. Also, indicate if each instruction performs an aligned memory access, an unaligned memory access, or no memory access at all.

Constant values in address calculations will be zero-extended out to 32 bits if necessary.

Initial state:

EAX: 00067340h
EBX: 11016FEBh
ECX: FFFFFFFDh
EDX: DEADBEEFh
ESI: 00067330h
EDI: 00000003h

Address	Lo			Hi
67330h	01	08	AD	C0
67334h	02	62	38	73
67338h	CE	12	60	EB
6733Ch	B0	55	99	DD
67340h	F4	88	22	0A

Instructions:

```
MOVZX  DX, BYTE PTR [ESI+03h]
```
Aligned? Yes No Not a memory access

```
MOV    EBX, 00067334h
```
Aligned? Yes No Not a memory access

```
XCHG   AX, [ESI+2*EDI]
```
Aligned? Yes No Not a memory access

```
LEA    EDI, [EBX+ECX]
```
Aligned? Yes No Not a memory access

```
MOVSX  ECX, WORD PTR [EAX]
```
Aligned? Yes No Not a memory access

3. (30 points) ***Arithmetic instructions***

For each instruction in the sequence shown below, list all changed registers and/or memory locations and their new values. If memory is changed, be sure to explicitly list **all changed bytes**. Where appropriate, you should also list the state of the carry flag (CF).

Constant values in address calculations will be zero-extended out to 32 bits if necessary.

Initial state:
EAX: 0000A30Fh
EBX: 00000700h
ECX: 000006FEh
EDX: 52DC7AFBh
CF: 0
ESI: 00011100h

Address	Lo			Hi
11800h	1B	79	02	10
11804h	00	44	15	5A
11808h	89	F6	A2	B1

Instructions:

```
ADD   [ESI+EBX], DX



SUB   EAX, ECX



DEC   WORD PTR [ESI+EBX+4]



NEG   BYTE PTR [ESI+EBX+000Ah]



IDIV  CH
```

4. (20 points) ***Logical instructions***

For each instruction in the sequence shown below, list all changed registers and/or memory locations and their new values. If memory is changed, be sure to explicitly list **all changed bytes**. Where appropriate, you should also list the state of the carry flag (CF).

Initial state:
EAX: 00000F6Dh
EBX: 0000FE9Ah
ECX: 00000006h
EDX: 000053DCh
CF: 0

Address	Lo			Hi
31700h	04	00	08	00
31704h	83	00	01	01
31708h	05	01	71	31
3170Ch	20	40	60	80
31710h	02	00	AB	0F

Instructions:

```
SAR    AX, 3
```

```
AND    AL, BL
```

```
SHL    AL, CL
```

```
NOT    DX
```

```
OR     AX, DX
```

5. (10 points) ***Extra credit***

Complete the code snippet below by writing the appropriate x86 instruction into each of the blank spaces. The purpose of each instruction is described in a comment to the right of the blank.

```
_____________________________________ ; Copy a word from
                                      ;  address 4370h into
                                      ;  the upper 16 bits of
_____________________________________ ;  EAX, using two
                                      ;  instructions. The
                                      ;  lower 16 bits of EAX
                                      ;  should be set to 0
_____________________________________ ; Divide the value in BX
                                      ;  by 128 and place the
                                      ;  result in BX, using a
                                      ;  single instruction
_____________________________________ ; Copy the sign bit of
                                      ;  the value in ECX
                                      ;  into the carry flag,
                                      ;  then ensure ECX holds
                                      ;  a negative value
_____________________________________ ; Move the value in the
                                      ;  carry flag into the
                                      ;  most significant bit
                                      ;  of DX (you may change
                                      ;  other bits of DX)
_____________________________________ ; Find the position of
                                      ;  the most significant
                                      ;  nonzero bit in BL,
                                      ;  and store that
                                      ;  position in CL
_____________________________________ ; Clear the middle 16
                                      ;  bits of EAX without
                                      ;  changing any other
                                      ;  bits in 1 instruction
_____________________________________ ; Store the sum of EAX,
                                      ;  EBX, and the constant
                                      ;  value 5 in ESI, using
                                      ;  1 instruction
_____________________________________ ; Multiply the byte at
                                      ;  the address stored in
                                      ;  ESI by 16, using 1
                                      ;  instruction
_____________________________________ ; Copy the rightmost bit
                                      ;  of EAX into CF
```