

Exercises For Exam

These are problems that covers all topics in this course. This exercise will not be graded. The sole purpose of this exercise is to help you preparing for the exam. There is **no guarantee** that these questions will be out in the exam.

Chapter 1

You can find the answer of these questions in the book.

1. Explain the difference of virtual machine and real machine.
2. In what sense virtual machine correspond to a real machine?
3. Why should we think the hypothetical machine into layers?
4. What is the relationship between layers, i.e. the upper layer with the layer(s) below it?
5. If there is a layer below digital logic level, what do you think the layer will be?
6. Which layer do you think machine language is at? Why?
7. If the programmer is allowed to program at microarchitecture level, what is the consequences?
8. Why operating systems is necessary?
9. What is the difference between timesharing and batch system?
10. What is the purpose of system call?
11. Explain the von Neumann machine. What is the advantages and disadvantages?
12. Compare von Neumann's architecture and bus architecture. Describe the advantages and disadvantages.

Try to answer exercise number 1, 2, 5, 6, 8, and 10 from the book.

Chapter 2

You can find the hint of these questions from the book.

1. How many register do you think a machine should have in minimum? Why?
2. Is there any memory to memory instruction? How can we deal with it?
3. Look into page 42 of your text book. If step 4 is omitted, what will happen?
4. Still on page 42. Explain the consequences if the memory address determined by step 4 is invalid?
5. Look into the program in page 43. There is a line: $PC = PC + 1$; We know that this line change the program counter to point to the next instruction. What does this statement imply?
6. Correspond to number 5, if the instruction length is variable, how should we modify the code and why? (Just describe the outline)
7. Is the interpreter mentioned in page 42 the same as interpreter mentioned in chapter 1? If it is the same in some sense, describe the difference between the two.

8. Why complex instruction set is born? (Hint: Page 42-44)
9. Correspond to number 8, why RISC is born?
10. Suppose we have a CISC and a RISC CPU. Both of them has the same specification and the same clock speed. If both CPU is then installed to computers with the same configuration and is doing the same task, which one do you think will perform faster?
11. Which memory type do you think is suitable for control stores mentioned in page 45? Why?
12. If in the future all memory accesses were very fast so that it will cause no penalty to CPU, which design principles can be ignored?
13. Describe the main differences between pipeline and superscalar architecture. (Hint: look into fig 2-4 and fig 2-5)
14. Why pipeline processors are suitable for highly sequential jobs?
15. Describe the main differences between multiprocessors and multicomputers.
16. Why array processors are suitable for highly parallel jobs?
17. How can we detect impossible codes in Hamming code?
18. Suppose we have the code 100101110000011011101. Detect whether it is correct or not. If it is wrong, mention which bit is wrong, then correct the code.
19. Why do we need cache memory? How does the cache work in general?
20. If miss ratio of a cache is high, what does it imply? How can we decrease miss ratio?
21. Look at figure 2-18. Briefly explain overall comparison from registers to tape / optical disks in the sense of: speed, capacity, price.
22. Which one do you think is faster: Hard drive or floppy drive? Why?
23. Do you think that all CD capacities used for data? Why? If all CD space is used for data, what is the consequences?
24. Suppose you are to design a video terminal. The terminal has 640x480 pixels. Each pixel can represent true color (i.e. 16,777,216 colors). To achieve a smooth video, the screen at least has to be repainted 30 times a second. How much is the bandwidth needed?
25. Suppose an engineer have built a system has 16,777,216 colors. However, the manager insists that it has to display only 65,536 for some model. The engineer argue that this is not efficient and he suggested to map only 256 colors or so. Why?

Try to answer exercise number 1 – 6, 8, 11, 14, 18, 19, 27 – 29.

Chapter 3 And Appendix A

You can find the hint of these questions from the book.

1. The truth table of $A \rightarrow B$ is defined as follows:

A	B	A @ B
1	1	1
1	0	0
0	1	1
0	0	1

- Construct the circuit for $A \rightarrow B$ using any gates
 - Same as part a, but only using NAND gates
 - Same as part a, but only using NOR gates. (Hint: Use De Morgan's Law).
- Repeat question number 1 with $(A \rightarrow B) \wedge (B \rightarrow A)$. (Hint: Construct the truth table first).
 - Construct a 4 bit subtractor out of 1-bit adders.
 - How can we get a finer resolution than basic clock?
 - Why DRAM is slower than SRAM?
 - Explain the process of subtraction in: 1's and 2's complement, and signed magnitude.
 - Do we have -128 in signed magnitude? In excess 128? Why?

Try to answer exercise number 4-6, 12, 13 from chapter 3 and bonus homework 1.

Chapter 7

You can find the hint of these questions from the book.

- Describe briefly why assembly language programming is difficult.
- Examine the following assembly instructions:

```

A EQU 5
B EQU 8
X EQU D*3          (1)
Y EQU 3
C EQU A*B          (2)
D EQU Y*B+C        (3)
:
:
I DW 0
J DW 1
K DB 1
L DB 0
:
:
MOV EAX, C          (4)
MOV EBX, Y*A        (5)
MOV ECX, EAX+D      (6)
MOV A, 4            (7)
MOV I, A+B*C        (8)
MOV J, I            (9)
MOV EAX, I          (10)
MOV J, EAX          (11)
MOV K, I            (12)

```

Look into the instruction with numbers.

- a. Specify which one is valid and which one is not. Explain.
- b. What is the result of EAX in instruction marked by number 10?
- c. What is the contents of the variables after executing number 11? (assume that all instructions run)

3. Explain the following assembly instructions. Is it legal? Why? What is the result of EAX?

```
A EQU B
B EQU A
MOV EAX, A
```

4. Examine the following assembly instructions:

```
DW 8          (1)
A DB 300      (2)
B DW 1000
C DD 500
ARRAY DD 3, 5, 2, 4, 1, 7, 6, 10, 8, 9    (3)
:
:
MOV EAX, B    (4)
MOV EBX, C    (5)
DB 200        (6)
MOV EDX, ARRAY+4 (7)
```

Look into the instruction with numbers.

- a. Specify which one is valid and which one is not. Explain.
- b. What is the contents of EDX after instruction number 7 is executed?

5. Consider the following macro definition:

```
SWAP MACRO X, Y
    MOV EAX, X
    MOV EBX, Y
    MOV X, EBX
    MOV Y, EAX
SWAP ENDM
```

Its high level language (HLL) version is as follows (in C-style pseudo code):

```
void swap (int x, int y)
{
    int temp;

    temp = x;
    x = y;
    y = x;
}
```

If we compare both versions, it seemed that the assembly language version is not efficient, i.e. it uses two temporary storages: `EAX` and `EBX`, while in HLL version only use `temp` as the temporary storage. If we modify the macro slightly as follows:

```
SWAP MACRO X, Y
    MOV EAX, X
    MOV X, Y
    MOV Y, EAX
SWAP ENDM
```

It is just similar to HLL. However, what are the consequences?

6. Look into the original `SWAP` macro in problem 5. Now, we implement it into the real code as follows (assume that the macro is already defined):

```
.DATA
    I DD 5
    J DW 3
    A DW 7
    K DD 4
    L DW 2
    B DW 8

.CODE
    SWAP I, K (1)
    SWAP J, L (2)
    SWAP A, L (3)
    MOV EDX, 8
    MOV ECX, 3
    SWAP EDX, ECX (4)
    MOV EAX, 7
    MOV ECX, 4
    SWAP ECX, EAX (5)
    MOV EAX, 10
    MOV EBX, 20
    SWAP EAX, EBX (6)
    HLT

END
```

- What is the contents of the variables after instruction number 1 is executed?
- The same as part a, but after instruction number 2.
- The same as part a, but after instruction number 3.
- What is the contents of the registers after the execution of number 4?
- The same as d, but after number 5.
- The same as d, but after number 6.

7. If you consider the advanced features in page 497, it has `IF`, `ELSE`, and `ENDIF`. Do you think it is an instruction? Explain.

8. Examine the following code:

```
A EQU 2*B
B EQU 5
```

```

START: MOV ECX, B
      MOV EAX, 0

L1:   CMP ECX, 0      ; Compare ECX with 0
      JE  L2         ; Jump to L2 if it is equal
      ADD EAX, ECX
      SUB ECX, 1     ; Subtract ECX by 1
      JMP L1

L2:   MOV RESULT, EAX
      HLT           ; Stop the program
      RESULT DD 0

```

- What is the content of EAX after the execution of the above snippet?
- Which label will cause forward reference? Why?
- Show the contents of symbol table after the first pass. We suppose that the label start is 1000 and each instruction takes 32-bits.
- Explain how pass 2 resolve the forward reference problem.
- Suppose we forgot to define B in the program above. Explain how the assembler will show the error.

9. What is the similarity and the difference between object modules and executable binary program.

10. Suppose we have a program that has 5 object modules M1, M2, ... M5, which sizes are 400, 300, 500, 700, and 200 respectively. We have the following public variables that are used throughout all modules

Variable name	Defined in	Original Address (Before Linking)
V1	M1	100
V2	M1	300
I	M2	50
J	M2	200
DATA	M3	150
TEMP	M3	350
ARRAY	M4	600
STATUS	M5	50

- Explain how the linking process of all modules
- What happened to public variables? Specify the new address after linking.
- If the resulting program is then loaded at address 400, will the program run? Why?

Try to answer questions in exam 5 and bonus homework 2

Chapter 4

1. How can we read and write the same register at one cycle?
2. Why are some registers only accessible at microarchitecture level?
3. What is the reason that MAR is not connected to B bus?
4. What is the reason that MBR is not connected to C bus?
5. How is the combination between MAR, MBR, and PC is used?

Try to answer question number 2 for Chapter 4.