시스템프로그램 중간시험 (2020학년도 겨울학기, 12/30/2020)

학과		학번		학년		이름		

(1) Fill in the blanks of the following sentences. Choose your answer in the Term-Box below.

○ The gcc compiler driver reads a source file and translates it into an executable object file. The translation is performed in the sequence of four phases and the programs that perform the four phases are preprocessor, compiler, (①), and linker.

○ Many modern processors have special hardware that allows a single instruction to cause multiple operations to be performed in parallel, a mode known as (②) parallelism.

○ Some machines choose to store the object in memory ordered from least significant byte to most, while other machines store them from most to least. The former convention, where the least significant byte comes first, is referred to as (③).

○ The storage devices in most computer systems are organized as a/an (④). As we move from the top of it to the bottom, the devices become slower, larger, and less costly per byte.

○ To inspect the contents of machine-code files, a class of programs known as disassemblers can be invaluable. These programs generate a format similar to assembly code from the machine code. With Linux systems, the program (⑤) can serve this role given the -d command-line flag.

○ For a binary number x = $(x_{w-1}, x_{w-2}, \cdots, x_0)$, its 1's complement encoding can be defined as

$$B2O_w(x) = (⑥ \qquad\qquad) + \sum_{i=0}^{w-2} x_i \cdot 2^i$$

○ IEEE FP format defines 4 different rounding modes. Among those, (⑦) is the default mode and attempts to find a closest match. For halfway values, it adopts the convention that it rounds the number either upward or downward such that the LSB of the result is zero.

○ There are several different encodings for jumps, while most commonly used ones are (⑧). They encode the difference between the address of the target instruction and the address of the instruction immediately following the jump.

○ A switch statement (in C language) provides a multi-way branching capability based on the value of an integer index. It is usually compiled with the (⑨), which is an array where each entry **i** is the address of a code segment implementing the action the program should take when the switch index equals **i**.

○ With IEEE floating-point standard, the largest normalized value has a bit representation with a sign bit 0, the least significant bit of the exponent equal to 0, and all other bits equal to 1. It has a significand value (⑩).

[Term-Box]

$1-2^{-n}$	symbol table	hyperthreading	round-toward-zero	assembler
$2-2^{-n}$	translator	piplining	round-to-even	objdump
SIMD	$-x_{w-1} \cdot (2^{w-1}+1)$	jump table	little endian	PC-relative
MISD	$-x_{w-1} \cdot (2^{w-1}-1)$	memory hierarchy	big endian	absolute

①	②	③	④	⑤
⑥	⑦	⑧	⑨	⑩

(2) Assume we are running code on a 6-bit machine using 2's-complement arithmetic for signed integers. A short integer is encoded using 3-bits. Fill in the empty boxes in the table below. You need not fill in entries marked "—". The following definitions are used in the table:

```
short sy = -2;
int y = sy;
int x = -10;
unsigned ux = x;
```

Expression	Decimal value	Binary representation
Zero	0	000000
—	10	001010
—	-20	⑥
ux	①	⑦
y + 3	②	⑧
x >> 3	③	⑨
-TMax	④	—
-TMin	—	⑩
TMax + TMin	⑤	—

(3) Consider the following 16-bit floating-point representation based on IEEE floating-point format.
- There is a sign bit in the MSB
- The next 7 bits are the exponent
- The last 8 bits are the significand

The rules are like those in the IEEE standard (normalized, denormalized, zero, infinity, and NaN), where we consider the floating-point format to encode numbers in a form

$$(-1)^s \times m \times 2^E$$

where m is the mantissa and E is the exponent.
Fill in the table below for the descriptions given, with the following instructions for each column.
- Hex: 4 hexadecimal digits
- m: The fractional value of the mantissa in the form of x/y, where x is an integer and y is an integral power of 2
- E: The integer value of the exponent
- Value: The numeric value represented, in the form of x * 2^z, where x and z are integers

Description	Hex	m	E	Value
Smallest value > 1	①	⑤	⑨	⑬
Largest denormalized	②	⑥	⑩	⑭
Largest normalized	③	⑦	⑪	⑮
-∞	④	—	—	—
Number 0x3AC0	3AC0	⑧	⑫	⑯

(4) For a function `decode2()`, **gcc** generates the following assembly code in the right side. Parameters x, y, and z are passed in registers %rdi, %rsi, and %rdx. The code stores the return value in register %rax. Fill in the C code for `decode2` that will have an effect equivalent to the assembly code shown.

`long decode2(long x, long y, long z)`
`{`
`long t1 = y - z;`
`long t2 = ①__________;`
`long t3 = (t1 << 63) >> 63;`
`long t4 = ②__________;`
`return t4;`
`}`
	`decode2:`
	`subq %rdx, %rsi`
	`imulq %rsi, %rdi`
	`movq %rsi, %rax`
	`salq $63, %rax`
	`sarq $63, %rax`
	`xorq %rdi, %rax`
	`ret`

①
②

(5) Consider the following source code, shown on the left side of the table, where **R**, **S**, and **T** are constants declared with **#define**. In compiling this program, **gcc** generates the following assembly code on the right side. Answer the following questions.

`long A[R][S][T];`
 	
`long code7(long i, long j, long k, long *d)`
`{`
`*d = A[i][j][k];`
`return sizeof(A);`
`}`
	`code7:`
	`leaq (%rsi,%rsi,2),%rax`
	`leaq (%rsi,%rax,4),%rax`
	`movq %rdi,%rsi`
	`salq $6,%rsi`
	`addq %rsi,%rdi`
	`addq %rax,%rdi`
	`addq %rdi,%rdx`
	`movq A(,%rdx,8),%rax`
	`movq %rax,(%rcx)`
	`movl $3640,%eax`
	`ret`

① Give a formula for the location of array element A[i][j][k], in terms of i, j, k, T, and S.
② Use your reverse engineering skills to determine the values of R, S, and T based on the given assembly code.

①
②

☆ 수고했습니다~.