Quiz-01 Solutions

(문) 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 ( ).
(답) little-endian

(문) ( ), sometimes called simultaneous multi-threading, is a technique that allows a single CPU to execute multiple flows of control. It involves having multiple copies of some of the CPU hardware, such as program counters and register files, while having only single copies of other parts of the hardware, such as the units that perform floating-point arithmetic.
(답) Hyperthreading

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

(문) The compiler+assembler reads a source code file, which is a kind of text file, and translates it into an ( ) object file.
(답) relocatable

(문) The translation by gcc compiler driver is performed in the sequence of 4 phases, and the programs that perform the 4 phases are ( ) in order.
(답) preprocessor – compiler – assembler - linker

Quiz-02 Solutions

(문) Convert a decimal integer 104 into 2-digit hexadecimal number.
(답) 0x68

(문) Convert a decimal integer -108 into 2-digit hexadecimal number in 2’s complement notation.
(답) 0x94

(문) What is the minimum value of the 12-bit signed integer in the system which uses 2's-complement encoding for negative integers.
(답) -2048

(문) Assume we are running code on a 6-bit machine using 2’s-complement arithmetic for signed integers. What is the value of ux in the following code segment.

$-2^5 \sim 2^5-1$ 6-bit 2's 2^6

-32 ~ 31 64 64

-24

```
---
int x = -24;
unsigned ux = x;
---
```

(답) 40

40

(문) Assume we are running code on a 5-bit machine using 2's-complement arithmetic for signed integers. What is the value of sum in the following code segment.

```
---
int x = 8;
int y = 12;
int sum;
sum = x + y;
---
```

-16 ~ 16

$20 - 2^5 = -12$

(답) -12

-12

Quiz-03 Solutions

$\frac{9}{8}$ 7 $\underline{0}\ \underline{0111}\ \underline{001}$

(문) In 8-bit floating-point format with 4 exponent bits and 3 fraction bits, what is the binary representation of decimal value 9/8?

(답) 00111001

00111001

(문) In 8-bit floating-point format with 4 exponent bits and 3 fraction bits, what is the decimal value of the binary representation 00111110?

7

(답) 7/4

$\frac{7}{4}$ $\underline{0}\ \underline{0111}\ \underset{4\,2\,1}{\overset{\frac{5}{8}}{\underline{110}}}$ $1 \times \frac{14}{8}$

(문) What is the smallest positive normalized value of the standard 64-bit double precision floating-point representation?

$1.0 \times 2^{e-bias}$ 1 11 52

(답) 1.0 × 2^(-1022)

e-bias 1.0×2^{-1022}

$1-(2^{11-1}-1) = 2-2^{10} = -1022$

(문) Which of the following is FALSE for floating-point numbers?

(답) Floating-point addition does not satisfy <u>monotonicity</u>.

(문) IEEE FP format defines 4 different rounding modes. Among those, (   ) 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 to be zero.

round to even

(답) round-to-even

Quiz-04 Solutions

(문) In x86-64 architecture, different registers serve different roles in typical programs.

most unique among them is the register (   ), used to indicate the end point in the run-time stack.
(답) %rsp

%rsp

(문) For the following line of assembly code, determine the appropriate instruction based on the operands provided.

```
   (           ) (%rsp,%rdx,8), %edx  → l
```

b w l q
l i e r
movl

(답) movl

(문) We have a function with prototype

```
void decode(long *xp, long *yp) {
   long x = *xp;
   long y = *yp;
   (                    )
}
```

which is compiled into assembly code, yielding the following:

```
decode:
   movq   (%rdi), %rcx        x = *xp
   movq   (%rsi), %rdx        y = *yp
   movq   %rdx, (%rdi)        *xp = y
   subq   %rcx, (%rsi)        *yp = y-x
   ret
```

*xp *yp

Fill in the C code of decode2.

(답) *xp = y; *yp = y-x;

(문) Suppose register %rbx holds value x and %rdx holds value y. Show the formula indicating the value that will be stored in register %rax for the following assembly instruction.

12+x+4y

```
   leaq 12(%rbx,%rdx,4), %rax
```

x y

(답) x + 4y + 12

(문) Consider the following function decode4.

```
short decode4(short x, short y, short z) {
   short t = (                          );
   return t;
}
```

which is compiled into assembly code, yielding the following:

rdi x rsi y rdx z

```
decode4:
   leaq (%rsi,%rsi,8), %rbx
   leaq (%rbx,%rdx), %rbx
   leaq (%rbx,%rdi,4), %rbx
   ret
```

y 8y 9y

9y z 9y+z

9y+z 4x 4x+9y+z

Fill in the missing C expression of decode4.

(답) 4x + 9y + z

Quiz-05 Solutions

(문) The following function foo1,

```
   int foo1(data_t x, data_t y) {
      return x COMP y;
   }
```

shows a general comparison between arguments x and y, where data_t is defined (via typedef) to be one of the integer data types and either signed or unsigned. The comparison COMP is defined via #define. For the following instruction sequence, determine which data type data_t and which comparison COMP could cause the compiler to generate this code.

```
   cmpl %esi, %edi
   setg %al
```

l - ei → 4B int

g → signed >

(답) data_t: int; COMP: >

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

PC-relative

(문) The following C function foo3,

```
   void foo3(short a, short *p) {
      if (a && *p < a)
```

```
        *p = a;
  }
```

is compiled into the following assembly code.

```
  foo3:
    (                    )          cmpq $1, %rdi
    je .L1                          testq %rdi, %rdi
            p       a
    cmpq (%rsi), %rdi
    jle .L1
    movq %rdi, (%rsi)
  .L1:
    rep; ret
```

What is the missing instruction in foo3?

(답) cmpq $0, %rdi

(문) The following assembly code computes n!, where the value of n is in register %rbx. What is the missing instruction?

```
  factorial4:
    movl $1, %eax
  .L4
    imulq %rbx, %rax
    (                    )      decq %rbx
    cmpq $1, %rbx
    jg .L4
    rep; ret
```

(답) decq %rbx

(문) When procedure P calls procedure Q, Q must preserve the values of the registers %rbx, %rbp, and %r12-%r15, ensuring that they have the same values when Q returns to P as they did when Q was called. These registers are called (   ) registers.

(답) callee-saved   callee-save

#