10/8 Quiz

Question 1

The RISC-V ISA is designed to have fixed-size instructions and to always require register operands in arithmetic instructions. Which design principle motivated this decision?

- Good design demands good compromises.
- Smaller is faster.
- Simplicity favors regularity.
- Make the common case fast.
- None of the above

Answer: Simplicity favors regularity.

Explanation:

每一個指令使用相同大小，會讓硬體設計更簡單與規則化。例如：不必去考慮下一個指令要從哪裡開始，每次執行指令後，PC 都是加 4 。指令解碼時也不用考慮指令長度。

算術指令的運算元 (operand) 放在暫存器 (register) 中，而不是像 x86 可以把運算元放在記憶體 (memory) ，甚至支援二個運算元都放在記憶體的情形，也是讓指令更簡單，例如： 加法指令就只有 add 和 addi 二種，而不是像 x86 加法指令就有多種情況。

這裡題目有個小瑕疵，實際上在 RISC-V 算術指令有 R-format 和 I-format 二種。像是 addi, ori 等，第二個運算元也可以放在指令本身 (Immediate) 。但即使這樣，RISC-V 設計也比其他 CISC 架構更簡單，故不影響本題的答案。

Question 2

What does this RISC-V instruction do?

```
sub a, b, c
```

- a = b - c
- a = c - b
- b = a - c
- b = c - a

Answer: a = b - c

Explanation:

在 RISC-V 中算術指令在組合語言表達的方式為

```
op rd, rs1, rs2
```

或

```
opi rd, rs1, imm
```

其中 `rd` 是儲存計算結果的暫存器，而 `rs1` 是第一個運算元， `rs2` 或 `imm` 是第二個運算元。

故本題對應到 C 語言的程式，即為 `a = b - c` 。

Question 3

How is -4 represented as an 8-bit signed integer in binary using the two's complement system? Fill in your answer using only 0s and 1s, without any spaces.

Answer: 11111100

Explanation:

用 2 的補數方式表示 -4 計算步驟為：

1. 先把 4 的 2 進位形式寫出來： `00000010`
2. 先計算 1 的補數。把每一個位元反轉，即為 `11111101`
3. 最後再加上 1 ，為 `11111100` ，就成為 2 的補數。

Question 4

How is the 32-bit integer 0x01234567 stored in memory on a little-endian, byte-addressed system?

- 01 23 45 67
- 67 45 23 01
- 76 54 32 10
- 45 67 01 23

Answer: 67 45 23 01

Explanation:

Little-Endian 代表 LSB 會先儲存，而 MSB 存在最後。故儲存的順序是 67 45 23 01 。

Question 5

Which RISC-V instruction(s) can be used to load the constant value 5 into register x6? Please select all possible answers.

- addi x6, x6, 5
- addi x6, x0, 5
- ori x6, x0, 5
- ori x6, x6, 5
- ld x6, 5
- None of the above

Answer:

- addi x6, x0, 5
- ori x6, x0, 5

Explanation:

x0 暫存器內容固定為 0 。故 `addi x6, x0, 5` (x6 = 0 + 5) 、 `ori x6, x0, 5` (x6 = 0 | 5) 等同把 x6 設為 5 (x6 = 5) 。

而其他答案 `addi x6, x6, 5` (x6 = x6 + 5) 和 `ori x6, x6, 5` (x6 = x6 | 5) 均無法達到把 x6 設為 5 的結果 (x6 = 5) 。 ld 指令為把記憶體內容讀取到暫存器，在此格式及用法就不對。