Solutions

Marking: **M1** a correct method, **A1** the correct answer, **B1** an independent mark.

Each answer shows the steps, then the **result**.

2.1

$32 - 6 - 3 \times 5 = \mathbf{11}$ bits

e.g. a function code that picks the exact operation, or a shift amount

2.2

-2^{11} to $2^{11} - 1$

-2048 to 2047

no: 3000 is out of range; use `lui` and `addi`

2.3

`addi x1, x2, 5`: immediate

`lw x1, 8(x2)`: base plus offset

`beq x1, x2, L`: PC-relative

2.4

RISC: fixed-length instructions; CISC: variable-length

RISC: only loads and stores touch memory; CISC: many instructions can use memory operands

3.1

(a) $2^3 = 8$, so **3** bits

(b) $16 - 3 \times 3 = \mathbf{7}$ bits; $2^7 = \mathbf{128}$ operations

(c) $16 - 4 - 3 - 3 = \mathbf{6}$ bits; -32 to 31

(d) `0000101 011 001 010`; `0000 1010 1100 1010` = `0x0ACA`

3.2

(a) $600 \times 2 + 400 \times 4 = \mathbf{2800}$ bytes; saving $1200/4000 = \mathbf{30}\%$

(b) instructions no longer all start on 4-byte boundaries, so fetch and decode need more hardware to find where each one starts

3.3

(a) 16 registers: $32 - 7 - 3 - 8 = \mathbf{14}$ bits; 32 registers: $32 - 7 - 3 - 10 = \mathbf{12}$ bits

(b) 14 bits: -8192 to 8191; 12 bits: -2048 to 2047

(c) some values must be kept in memory (spilled); each use adds a load or store, which costs time and energy

exercises.lol exercises.com.cn

(d) 32 registers: loops with many live values avoid spills, and the 12-bit immediate is enough for most constants and offsets, while `lui` handles the rare large one