3.4 Machine instructions

Name: _____ Class: _____ Date: _____ **Total: 24 marks**

KEY WORDS addressing mode

Objective

Read and write small RISC programs.

1 Key knowledge and worked examples

■ **The shapes**

`add rd, rs1, rs2` · `addi` · `lw`/`sw` with base+offset · `beq`/`bne` ✓ – memory only via load/store; every operand otherwise a register or immediate.

■ **Worked example: C to instructions**

`a[i] = a[i] + 1` → `lw` · `addi` · `sw` (index scaled into the offset) ✓; a while loop → `addi R1, R1, -1` + `bne R1, R0, loop` – two instructions per iteration.

2 Practice

2.1 In `lw R1, 8(R2)`, what is the 8? [3]

2.2 Which instruction family touches memory? [3]

2.3 Write the pair that counts R1 down and loops while nonzero. [3]

3 Exam-style questions

3.1 Assemble: sum the five integers at mem[200..216] into R5.

Write the instruction sequence, naming any labels you use. State the register that serves as the loop counter and the one as the running sum. Give the iteration count and the final offset on R2. [8]

3.2 Map C constructs to instruction shapes.

For each of `while (n > 0)`, `a[i] = a[i] * 2`, and `if (x == y) goto L`, give the instructions it compiles to. Name the addressing mode of every operand. Explain in one line why keeping arithmetic register-only helps pipelining. [7]