LSP Exam Results & Solutions — January 13, 2026

CVUT FEL (CV̌UT) – Cěské vysoké učění technické v Praze | Czech Technical University in Prague

中文版 | English | Cěsťina

Course: B0B35LSP – Logické systémy a procesory | BE5B35LSP – Logic Systems and Processors

Keywords: Zkousǩa, Exam, Test, Solutions, Výsledky, Answers, K-Map, RS Latch, Pipeline

---

Results (Official) + Detailed Solutions

Source PDF (official sheet): 2026-01-13_Exam_Results_Official.pdf

Notes: The PDF contains diagrams (circuits/K-maps). Text extraction preserves most official answers, but some purely-graphical parts (notably Q6/Q7 drawings) may not appear as plain text. When a diagram answer is not fully extractable, this document links back to the PDF and provides a robust solving method.

Q1 — Moore/Mealy FSM ordered tuple

Official answer (from PDF)

- X is **finite set of all input vectors**
- S is **finite set of all output vectors**
- Z is **finite set of all internal states**
- δ is a mapping
  - Moore: $\delta : X \times S \to S$
  - Mealy: $\delta : X \times S \to S$
- ω is a mapping
  - Moore: $\omega : S \to Z$
  - Mealy: $\omega : X \times S \to Z$
- s_0 is **initial state** $s_0 \in S$

Explanation (how to write it fast)

- **Moore**: output depends only on state S.
- **Mealy**: output depends on input+state $X \times S$.

Q2 — Simulate circuit outputs X and Y at $t_0..t_4$

Official answer (from PDF)

Inputs (as shown): – A: `0 | 1 | 1 | 0 | 0` – B: `1 | 1 | 0 | 0 | 1`

Outputs: – X: `0 | 1 | 1 | 1 | 0` – Y: `1 | 0 | 0 | 0 | 1`

Explanation (why these values)

From the official Q3 formula for the same circuit (see next question), we can interpret X as a **level-sensitive latch** controlled by B: – If $B = 1$ then X follows A (transparent) – If $B = 0$ then X holds its previous value (memory)

So: – t_0: $B = 1 \Rightarrow X = A = 0$; and the given result shows $Y = \neg X = 1$ – t_1: $B = 1 \Rightarrow X = A = 1; Y = 0$ – t_2: $B = 0 \Rightarrow X$ holds $1; Y = 0$ – t_3: $B = 0 \Rightarrow X$ still holds $1; Y = 0$ – t_4: $B = 1 \Rightarrow X = A = 0; Y = 1$

Q3 — Shannon expansion + Karnaugh maps (f_0, f_1)

Official answer (from PDF)

The official expression shown:

- $$X = (A \land B) \lor (X \land (\neg B \lor (A \land B)))$$

And: – $f_0 = A \land B$ – $f_1 = (A \land B) \lor (\neg B \lor (A \land B)) = (A \land B) \lor \neg B = A \lor \neg B$

Explanation (copy-paste method)

Given $X = f(A, B, X)$, Shannon on variable X:

$$X = (\neg X \land f_0(A, B)) \lor (X \land f_1(A, B))$$

Compute by substitution: – $f_0(A, B) = f(A, B, 0)$ – $f_1(A, B) = f(A, B, 1)$

Then simplify: – $f_1 = (A \land B) \lor \neg B = A \lor \neg B$ (absorption)

Interpretation shortcut: – When $B = 1$: $f_1 = A \lor 0 = A$ so X becomes A (transparent) – When $B = 0$: $f_1 = A \lor 1 = 1$ so X becomes X (hold)

Q4 — 12-bit number as unsigned and signed (two's complement)

Number: `1000 0001 1111`

Official answer (from PDF)

- Unsigned: **2079**
- Signed (two's complement): **−2017**

Explanation

Let $N = 12$, unsigned value: – $1000\ 0001\ 1111_2 = 2^{11} + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 2048 + 31 = 2079$

Two's complement signed: – MSB is 1 so it is negative –

$$\text{signed} = \text{unsigned} - 2^{12} = 2079 - 4096 = -2017$$

Q5 — Equivalent logic functions

The PDF text extraction shows the four candidate functions ($y_1 .. y_4$), but does not explicitly print the "marked" selection.

Verified equivalence (by exhaustive truth-table check)

- $y_1 \equiv y_2 \equiv y_4$
- y_3 **is not equivalent to them**

Truth tables (ordered by A, B, C from `000` to `111`): – y_1: `11011000` – y_2: `11011000` – y_4: `11011000` – y_3: `01011100`

So the correct "mark" set is: y_1, y_2, y_4.

Q6 — One-bit full adder (complete the schema)

The PDF page shows a schematic to be completed; the exact gate drawing may not appear in plain text extraction.

Standard correct completion (equations)

Let inputs be A, B, C_{in} and outputs be Sum S and Carry-out C_{out}:

- $$S = A \oplus B \oplus C_{in}$$
- $$C_{out} = (A \wedge B) \vee (C_{in} \wedge (A \oplus B))$$

Wiring hint

- First XOR: $X_1 = A \oplus B$
- Second XOR: $S = X_1 \oplus C_{in}$
- ANDs: $G_1 = A \wedge B, G_2 = C_{in} \wedge X_1$
- OR: $C_{out} = G_1 \vee G_2$

Q7 — Gate-only implementation (AND/NAND/OR/NOR/NOT)

The official result for Q7 is primarily diagrammatic in the PDF. If you need an exact "final wiring", refer to the diagram in: – 2026-01-13_Exam_Results_Official.pdf

Reliable solving method

- Rewrite everything into only **NAND** (or only **NOR**) using De Morgan
- Replace XOR/XNOR using standard NAND/NOR constructions
- Keep track of inversions (bubble-pushing)

Q8 — 5 pipeline instruction phases + operations

Official answer (from PDF)

- **FETCH** — fetches an instruction from memory.
- **DECODE** — the instruction is decoded and operand values are read from registers.
- **EXECUTE** — ALU performs an operation on decoded operands.
- **MEMORY** — possible work with memory; read/write using ALU address; if branch then write new address to PC.
- **WRITE-BACK** — results are saved in registers.

Exam-friendly wording

- IF / ID / EX / MEM / WB (the classic 5-stage pipeline naming)

Appendix — How Q5 equivalence was checked

A brute-force truth table over all $A, B, C \in \{0, 1\}$ shows $y_1 = y_2 = y_4$ for all 8 combinations.