2025-06-09 LSP Exam Solution / Rěsění zkousǩy / 考试解析

Course: B0B35LSP – Logické systémy a procesory | BE5B35LSP – Logic Systems and Processors **University**: CTU FEL (CTU) – Czech Technical University in Prague **Keywords**: Zkousǩa, Exam, Test, Solutions, Vysledky, Answers, K-Map, RS Latch, Pipeline

CN Version | EN Version | CZ Version

LSP Exam — 2025-06-09

Exam info

- Date: 2025-06-09
- Language: Czech (source)
- Type: standard exam
- Official answers verified from PDF

Q1 — RS latch simulation (5)

Task: Inputs A, B, C have the values shown at times $t_0..t_4$. Write the values of outputs X and Y.

Input sequence:

```
A = 1 | 1 | 1 | 0 | 1
B = 1 | 1 | 0 | 1 | 1
C = 0 | 1 | 0 | 0 | 0
    t0  t1  t2  t3  t4
```

Official answer:

```
X = 0 | 0 | 1 | 1 | 0   (binary: 00110)
Y = 1 | 0 | 0 | 0 | 1   (binary: 10011)
```

Quick reasoning sketch: – Step through $t_0..t_4$ and apply the RS latch behavior according to the circuit on the sheet (reset/set priority matters).

Q2 — Shannon expansion (6)

Task: Decompose the feedback function $X = f(A, B, C, X)$ from Q1 into:

$$X = (\neg X \wedge f_0(A, B, C)) \ \vee \ (X \wedge f_1(A, B, C)).$$

Official answer (K-map shown on sheet):

```
f0:     B                   f1:     B
      A  0  1                     A  0  1
C 0   0  0  1  0     C 0   1  0  1  0
  1   1  0  0  0       1   1  0  1  0
```

Official expression (as given):

```
X = (not C and B and A) or (C and not B and not A)
    or (X and not B and not A) or (X and B and A)
```

Q3 — Equivalent logic functions (4)

Task: Check all logic functions that have another equivalent function.

```
y1 <= (A or D) and (not A or C);
y2 <= C or (A and C and B) or (not A and C and D);
y3 <= (not A and D) or (A and not D) or (C and D);
y4 <= (C and D) or (not A xor not D);
```

Official answer: $y3 \equiv y4$.

Q4 — 8-bit register arithmetic (2)

Task: Store the lower bits of $124 + 125 + 126 + 127$ into a 1-byte register; what is the decimal value as an 8-bit number?

Computation:

```
124 + 125 + 126 + 127 = 502
502 mod 256 = 246
```

Answer: – Unsigned: 246 – Signed: -10 (since $246 - 256 = -10$)

Q5 — Moore/Mealy automaton definition (3)

Task: Complete the definition (mathematically precise).

Automat Moore (Mealy) is an ordered 6-tuple:

$$M = \langle X, S, Z, \omega, \delta, s_0 \in S \rangle.$$

- X is a finite input alphabet
- S is a finite set of states
- Z is a finite output alphabet
- δ is the state transition function:
  - Moore: $\delta : S \times X \to S$
  - Mealy: $\delta : S \times X \to S$
- ω is the output function:
  - Moore: $\omega : S \to Z$
  - Mealy: $\omega : S \times X \to Z$
- s_0 is the initial state

Q6 — +1 adder design (7)

Task: Implement a +1 incrementer using gates (no full adder).

Typical 4–bit incrementer structure: – $s_0 = \neg x_0$ – $s_1 = x_1 \oplus x_0$ – $s_2 = x_2 \oplus (x_0 \wedge x_1)$ – $s_3 = x_3 \oplus (x_0 \wedge x_1 \wedge x_2)$ – $carry = x_0 \wedge x_1 \wedge x_2 \wedge x_3$

Q7 — VHDL bit manipulation (7)

Task: Describe the circuit using concurrent VHDL, without sequential statements. Full points for a single statement in the architecture.

```vhdl
library ieee; use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
entity Test20250609q7 is
  port(x: in std_logic_vector(3 downto 0);
       y: out std_logic_vector(3 downto 0));
end entity;
architecture rtl of Test20250609q7 is
begin
  y <= x and (x(2 downto 0) & '1');
end architecture rtl;
```

Q8 — VHDL circuit analysis (7+1)

Task: Draw the block diagram for the circuit described by the following code (use symbols for gates, MUX, +1 adder, comparator, DFF registers).

```vhdl
library ieee; use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
entity Ex20250609q7 is
  port (CLK : in std_logic; Q : out std_logic);
end entity;
architecture rtl1 of Ex20250609q7 is
begin
  iproc: process(CLK)
    constant M: integer := 9;
    variable cntr : integer range 0 to M := 0;
    variable x : std_logic := '0';
  begin
    if rising_edge(CLK) then
      if cntr < M then
        cntr := cntr + 1;
      else
        cntr := 0;
        x := not x;
      end if;
    end if;
    Q <= x;
  end process;
end architecture;
```

Interpretation: frequency divider / modulo–10 counter with output toggle.

Q9 — Branch prediction (8)

Note: your CN master marks this as “not on exam” for 2026–01 context, but the original task is included here for completeness.

Task: For the following code, assuming the `for` loop is compiled to `do-while`, compute the number of branch mispredictions for the given predictors.

```
int data[] = { 0, 1, -2, -3, 4, -5, 6, -7, -8, 9 };
int min = INT_MAX;
for (int i = 0; i < 10; i++) {
  if (data[i] < min) min = data[i];
}
```

Official answer (as written in CN master): – 1–bit predictor (initial NT): misses = 9 (if–branch 7 + loop 2) – 2–bit predictor (initial WT): misses = 7 (if–branch 6 + loop 1)

Branch outcome sketch (from sheet notes):

```
Branch: N  T  N  N  T  N  T  N  N  T
Data:   0  1 -2 -3  4 -5  6 -7 -8  9
```