

2025–06–04 LSP Exam Solution / Rěšení zkousky / 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:** Zkouska, Exam, Test, Solutions, Vysledky, Answers, K–Map, RS Latch, Pipeline

[CN Version](#) | [EN Version](#) | [CZ Version](#)

Exam info

- Date: 2025–06–04
 - Language: Czech (source), with official answers verified from PDF
-

Q1 — RS latch simulation (5)

Given input waveforms at $t_0 \dots t_4$:

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

Official answer: – X = 01101 ($t_0=0, t_1=1, t_2=1, t_3=0, t_4=1$) – Y = 11000 ($t_0=1, t_1=1, t_2=0, t_3=0, t_4=0$)

How to reason quickly: identify which input drives **Reset** (here: A) and which drives **Set** (here: B·C), then step through times assuming gate delays are negligible.

Q2 — Shannon expansion (8)

Goal: decompose the feedback function $X = f(A, B, C, X)$ 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 1 1 1	C	0	1 1 1 1
1	0 1 1 1	1	0 1 1 1	1	0 1 1 1

Simplified expression (as given):

$$X = (X \text{ and not } C) \text{ or } A \text{ or } B$$

Method: compute $f_0 = f(A, B, C, 0)$ and $f_1 = f(A, B, C, 1)$ by substitution, then minimize with Karnaugh maps.

Q3 — Equivalent logic functions (4)

```
x1 <= (B and not A) or (A and not B);  
x2 <= (A and not C) xor (C and A);  
x3 <= (B or A) and (not B or not A);  
x4 <= (C xor A) or (B and not A);
```

Official answer: $x1 \equiv x3$ (both equal $A \oplus B$).

Q4 — 9-bit register arithmetic (2)

Question: store the lower bits of 4×510 into a 9-bit register.

Official result: – Unsigned: 504 – Signed (two's complement): –8

Reason: $-4 \cdot 510 = 4 \cdot (2^9 - 2) = 2^{11} - 8$. – Lower 9 bits $\equiv 2^9 - 8 = 512 - 8 = 504$. – Signed: $504 - 512 = -8$.

Q5 — 1-bit full adder (6)

A correct schematic implements:

```
Sum = A XOR B XOR Carry_in  
Carry_out = (A AND B) OR (Carry_in AND (A XOR B))
```

Q6 — 4:1 MUX in VHDL (4+4)

Concurrent statement:

```
qcon <= z when a1='1' else y when a0='1' else x;
```

Sequential statement:

```
process(all)  
begin  
    if a1='1' then qseq <= z;  
    elsif a0='1' then qseq <= y;  
    else qseq <= x;  
    end if;  
end process;
```

Q7 — Shift register analysis (8+2)

```
library ieee; use ieee.std_logic_1164.all;  
entity Test20250604q7 is  
    port (A, B : in std_logic;  
          C : in std_logic_vector(3 downto 0);  
          D : out std_logic);  
end entity;
```

```

architecture rtl of Test20250604q7 is
begin
  process(A)
    variable rg: std_logic_vector(C'RANGE);
  begin
    if rising_edge(A) then
      if B='1' then rg := C;
      else rg := rg(2 downto 0) & not rg(3);
      end if;
    end if;
    D <= rg(0);
  end process;
end architecture;

```

Correct name: ring shift register with inverted feedback — **Johnson counter**.

Key behavior: – Update on rising edge of A – If B=1: load rg := C – Else: shift and append NOT of MSB

Example sequence: 0000→1000→1100→1110→1111→0111→0011→0001→0000

Q8 — Direct-mapped cache (not required for 2026 scope)

This topic is marked as not required for the 2026–01 exam scope; kept here for completeness.

(See CN version for the full mapping table and hit/miss trace.)