

LSP Ultimate Review Guide / Ultimátní průvodce opakováním / 终极复习指南

Course: B0B35LSP – Logické systémy a procesory | BE5B35LSP – Logic Systems and Processors University: CVUT FEL (CVUT) – České vysoké učení technické v Praze | Czech Technical University in Prague Keywords: Zkouska, Exam, Test, Solutions, Vysledky, Answers, K-Map, RS Latch, Pipeline

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

LSP 考试终极复习指南 – 10小时冲刺版

考试时间: 2026年1月13日 10:00–11:30
考试地点: Karlovo náměstí A楼 KN-A-310
题目数量: 8道题
最后更新: 2026年1月13日 (考试当天)

考试范围确认: 根据老师邮件, 不考分支预测和Cache, 会考流水线!

目录

1. 有符号/无符号数计算
 2. 等价逻辑函数判断
 3. RS锁存器电路仿真
 4. 香农展开
 5. RS锁存器画图
 6. VHDL代码分析与编写
 7. Moore/Mealy自动机定义
 8. 分频器与计数器设计
 9. 本次不考
 10. Cache 本次不考
-

1 有符号/无符号数计算 必考

题型

给定n位二进制数, 求其无符号和有符号(补码)的十进制值。

公式速记

无符号数 (Unsigned):

$$= \sum_{i=0}^{n-1} b_i \times 2^i$$

有符号数 (Two's Complement):

$$= -b_{n-1} \times 2^{n-1} + \sum_{i=0}^{n-2} b_i \times 2^i$$

经典例题与答案

例1: 10位 10 0000 1111

$$\begin{array}{cccccccccc} : & 9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\ : & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}$$

$$\begin{aligned} : 2^9 + 2^3 + 2^2 + 2^1 + 2^0 &= 512 + 8 + 4 + 2 + 1 = 527 \\ : -2^9 + 15 &= -512 + 15 = -497 \end{aligned}$$

答案: 无符号 = 527, 有符号 = -497

例2: 12位 1000 0000 0111

$$\begin{aligned} : 2^{11} + 2^2 + 2^1 + 2^0 &= 2048 + 4 + 2 + 1 = 2055 \\ : -2^{11} + 7 &= -2048 + 7 = -2041 \end{aligned}$$

答案: 无符号 = 2055, 有符号 = -2041

例3: 8位存储 $124+125+126+127$ 的结果

$$\begin{array}{l} : 124+125+126+127 = 502 \\ 502 : 1 \ 1111 \ 0110 \ (9) \\ 8 : 1111 \ 0110 \end{array}$$

$$\begin{aligned} : 246 \\ : -10 \end{aligned}$$

答案: 无符号 = 246, 有符号 = -10

例4: 12位存储 1023^2 的结果

$$\begin{aligned} 1023^2 &= 1046529 \\ : 1111 \ 1111 \ 1100 \ 0000 \ 0001 \\ 12 : 0000 \ 0000 \ 0001 &= 1 \\ : 1 \\ : 1 \end{aligned}$$

快速计算技巧

1. 无符号: 直接按权重加
2. 有符号: 最高位为1时, 用 $-2^{(n-1)}$ +
3. 或者: 取反加1得到绝对值, 然后加负号

2 等价逻辑函数判断 必考

题型

给定多个逻辑函数，找出哪些函数是等价的。

解题方法：用卡诺图！

1. 对每个函数画卡诺图
2. 比较卡诺图，完全相同的函数等价

经典例题（每年必考！）

```
f1 <= (A xor C) or (A and not C);  
f2 <= (B or C) and (not A or B or C);  
f3 <= ((C and not B) or (B and A));  
f4 <= (A or C) and (not A or not C);  
f5 <= (A and not B) xor (A and C);  
f6 <= (A and not C) or (C and not A);
```

详细分析

$$f4 = (A \text{ or } C) \text{ and } (\text{not } A \text{ or } \text{not } C)$$

$$\begin{aligned} &= (A + C) \cdot (\bar{A} + \bar{C}) \\ &= A \cdot \bar{A} + A \cdot C + C \cdot \bar{A} + C \cdot \bar{C} \\ &= A \cdot C + \bar{A} \cdot C \\ &= A \oplus C \quad (\text{XOR}) \end{aligned}$$

卡诺图：

$$\begin{array}{cc} C=0 & C=1 \\ \begin{array}{cc} A=0 & 0 \\ A=1 & 1 \end{array} & \begin{array}{c} 1 \\ 0 \end{array} \end{array}$$

$$f6 = (A \text{ and } \text{not } C) \text{ or } (C \text{ and } \text{not } A)$$

$$\begin{aligned} &= A \cdot C + C \cdot \bar{A} \\ &= A \oplus C \quad (\text{XOR}) \end{aligned}$$

卡诺图（与f4相同）：

$$\begin{array}{cc} C=0 & C=1 \\ \begin{array}{cc} A=0 & 0 \\ A=1 & 1 \end{array} & \begin{array}{c} 1 \\ 0 \end{array} \end{array}$$

答案: $f4 \equiv f6$ (都是 $A \oplus C$)

常见等价形式速记

- $A \text{ XOR } C = A \cdot C + \bar{A} \cdot \bar{C} = (A+C) \cdot (\bar{A}+\bar{C})$
- $A \text{ XNOR } C = A \cdot C + \bar{A} \cdot \bar{C} = (A+C) \cdot (\bar{A}+\bar{C})$

3 RS锁存器电路仿真 必考

题型

给定含RS锁存器的电路和输入时序，求输出序列。

RS锁存器真值表

NOR型 RS锁存器（高电平有效）

S	R	Q	Q̄	说明
0	0	Q	Q̄	保持
0	1	0	1	复位
1	0	1	0	置位
1	1	?	?	禁止

NAND型 RS锁存器（低电平有效）

S̄	R̄	Q	Q̄	说明
1	1	Q	Q̄	保持
1	0	0	1	复位
0	1	1	0	置位
0	0	1	1	禁止

经典例题 (2025年6月9日考试)

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

分析步骤: 1. 分析电路结构，找出RS锁存器的S和R输入 2. 根据输入计算每个时刻的S和R 3. 查表确定Q输出

答案:

X = ...1..|..1..|..0..|..0..|..0..|
Y = ...0..|..0..|..1..|..0..|..1..|

经典例题 (2025年6月19日考试)

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

答案:

X = ...0..|..0..|..0..|..0..|..1..|
Y = ...1..|..0..|..1..|..1..|..0..|

解题技巧

1. 先画出电路的逻辑表达式
 2. 识别S和R的来源
 3. 逐个时刻分析，注意保持状态
-

4 香农展开 必考

题型

将函数 $Q = f(A, B, C, Q)$ 分解为：

$$Q = (\text{not } Q \text{ and } f_0(A, B, C)) \text{ or } (Q \text{ and } f_1(A, B, C))$$

香农展开公式

$$f(x_1, \dots, x_n, Q) = \bar{Q} \cdot f(x_1, \dots, x_n, 0) + Q \cdot f(x_1, \dots, x_n, 1)$$

即： $-f_0 = f(A, B, C, 0)$ — 令 $Q=0$ 代入 $-f_1 = f(A, B, C, 1)$ — 令 $Q=1$ 代入

经典例题详解

原函数： $Q = (A \cdot B) \cdot (Q + (B \cdot C))$

Step 1: 计算 f_0 (令 $Q=0$)

$$\begin{aligned} f_0 &= f(A, B, C, '0') \\ &= (A \cdot B) \cdot ('0' + (B \cdot C)) \\ &= (A \cdot B) \cdot (B \cdot C) \end{aligned}$$

Step 2: 计算 f_1 (令 $Q=1$)

$$\begin{aligned} f_1 &= f(A, B, C, '1') \\ &= (A \cdot B) \cdot ('1' + (B \cdot C)) \\ &= (A \cdot B) \cdot 1 \\ &= (A \cdot B) \end{aligned}$$

Step 3: 画卡诺图

$f_0 = (A \equiv B) \cdot (B \neq C)$ 的卡诺图：

	C=0	C=1
AB=00	0	1
AB=01	0	0
AB=11	1	0
AB=10	0	0

$f_1 = (A \equiv B)$ 的卡诺图：

	C=0	C=1
AB=00	1	1
AB=01	0	0
AB=11	1	1
AB=10	0	0

2025年6月9日考题答案

f0卡诺图:

C=0	C=1
AB=00	0
AB=01	1
AB=11	1
AB=10	0

f1卡诺图:

C=0	C=1
AB=00	1
AB=01	1
AB=11	1
AB=10	0

解题步骤

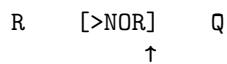
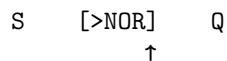
- 写出电路的逻辑表达式 $Q = f(A, B, C, Q)$
 - 令 $Q=0$, 化简得 f_0
 - 令 $Q=1$, 化简得 f_1
 - 分别画卡诺图
-

5 RS锁存器画图 (NOR/NAND)

题型

用纯NOR门或纯NAND门画出RS锁存器。

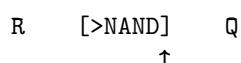
NOR型 RS锁存器



VHDL描述:

```
Q  <= S nor Qn;  
Qn <= R nor Q;
```

NAND型 RS锁存器



注意: NAND型输入是低电平有效!

真值表 (2025年6月19日考题)

NOR型: | R | S | Q | QN | |---|---|---|---| 0 | 0 | mem | mem |

NAND型: | R | S | Q | QN | |---|---|---|---| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |

6 VHDL代码分析与编写

常见题型

1. 分析给定VHDL代码，画出电路图
2. 补全VHDL代码实现指定功能

经典例题1: 4位移位寄存器

原始代码 (格式错误) :

```
library IEEE; use IEEE.STD_LOGIC_1164.all;
entity test is port (a,b,c,d:in std_logic; e:out std_logic); end;
architecture rtl of test is begin process(a,b) variable z:std_logic_vector(0 to 3); begin
if b='0' then z:=(others=>'0'); elsif rising_edge(a) then
if c='1' then z:=d & z(0 to 2); else z:=z(3) & z(0 to 2); end if; end if; e<=z(3); end process; end rtl;
```

格式化后:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity test is
    port (a, b, c, d : in std_logic;
          e : out std_logic);
end;

architecture rtl of test is
begin
    process(a, b)
        variable z: std_logic_vector(0 to 3);
    begin
        if b = '0' then
            z := (others => '0');      --
        elsif rising_edge(a) then
            if c = '1' then
                z := d & z(0 to 2);  --    d
            else
                z := z(3) & z(0 to 2); --
            end if;
        end if;
        e <= z(3);                  --
    end process;
end rtl;
```

功能: 4位可控双模式移位寄存器 – a: 时钟 – b: 异步清零 (b='0'时清零) – c: 模式选择 (c='1'串行输入, c='0'循环移位)
– d: 数据输入 – e: 输出z(3)

经典例题2: 100位移位寄存器

```
library IEEE; use IEEE.STD_LOGIC_1164.all;
entity pos100 is
    port (clock, d, sclrn : in std_logic;
          q: out std_logic);
end pos100;

architecture rtl of pos100 is
    signal reg : std_logic_vector(99 downto 0);
begin
    process(clock)
    begin
        if rising_edge(clock) then
            if sclrn = '0' then
                reg <= (others => '0'); --
            else
                reg <= d & reg(99 downto 1); --
            end if;
        end if;
    end process;
    q <= reg(0); -- 100
end rtl;
```

经典例题3: D触发器

```
library IEEE; use IEEE.STD_LOGIC_1164.all;
entity mydff is
    port (clock, d, aclrn, sclrn : in std_logic;
          q : out std_logic);
end mydff;

architecture rtl of mydff is
begin
    process(clock, aclrn)
    begin
        if aclrn = '0' then
            q <= '0'; --
        elsif rising_edge(clock) then
            if sclrn = '0' then
                q <= '0';
            else
                q <= d;
            end if;
        end if;
    end process;
end rtl;
```

2025年6月9日考题: XOR移位电路

```
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 <= ('0' & x(3 downto 1)) xor x;
end architecture rtl;

-- :
-- y <= x(3) & (x(3 downto 1) xor x(2 downto 0));
```

2025年6月19日考题: 16路解复用器

```
library ieee; use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity Test20250619q6 is
    port(data : in std_logic;
          addr: in std_logic_vector(3 downto 0);
          x: out std_logic_vector(0 to 15));
end entity;

architecture rtl of Test20250619q6 is
begin
    process(data, addr)
    begin
        x <= (others => '0');
        x(to_integer(unsigned(addr))) <= data;
    end process;
end architecture;
```

12分频器 (50%占空比)

```
library ieee; use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity div12 is
    port (clk : in std_logic;
          q12: out std_logic);
end entity;

architecture rtl of div12 is
    signal cntr : integer range 0 to 5 := 0;
    signal q_int : std_logic := '0';
begin
    process(clk)
    begin
        if rising_edge(clk) then
```

```

        if cntr = 5 then
            cntr <= 0;
            q_int <= not q_int; -- 6
        else
            cntr <= cntr + 1;
        end if;
    end if;
end process;
q12 <= q_int;
end architecture;

```

7 Moore/Mealy自动机定义

题型

填写自动机的数学定义。

标准定义

Moore/Mealy自动机是一个六元组 $M = \langle X, S, Z, \omega, \delta, s_0 \in S \rangle$

符号	含义
X	有限的输入向量集合 (finite set of all input vectors)
S	有限的内部状态集合 (finite set of all internal states)
Z	有限的输出向量集合 (finite set of all output vectors)
δ	状态转移函数 $\delta: X \times S \rightarrow S$
ω	输出函数
s_0	初始状态 $s_0 \in S$

Moore vs Mealy 区别

	Moore	Mealy
输出函数 ω	$\omega: S \rightarrow Z$	$\omega: X \times S \rightarrow Z$
输出依赖	仅依赖当前状态	依赖当前状态和输入

8 分频器与计数器设计

+1加法器（不用全加器）

题目：用逻辑门实现4位+1加法器

x0 [NOT] s0

x1 [XOR] s1

x2 [XOR] s2

x3 [XOR] s3

[AND] s4 (carry)

逻辑表达式:

```
s0 = not x0  
s1 = x1 xor x0  
s2 = x2 xor (x1 and x0)  
s3 = x3 xor (x2 and x1 and x0)  
s4 = x3 and x2 and x1 and x0 --
```

20分频器 (50%占空比)

2025年6月9日考题: 画出分频器电路图

```
CLK → [ 0~9 ] → [ <9 ] → [MUX] → [DFF] → Q  
          ↓           ↑  
          [+1 ]  
  
=9 Q
```

功能: 每10个时钟翻转一次输出, 实现20分频, 50%占空比

18分频器 (异步)

使用5个DFF级联, 检测到17(10001)时复位: - 用AND门检测 $Q_4 \cdot Q_0 = 1$ - 连接到所有DFF的异步清零端

9 跳转预测器 本次不考

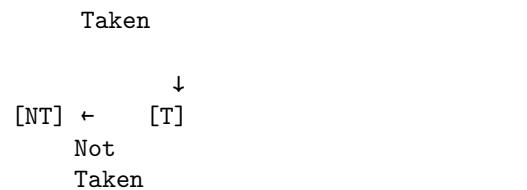
非考点提示: 根据2026年1月考试说明, 分支预测器本次不考, 可战略性跳过此章节。

题型

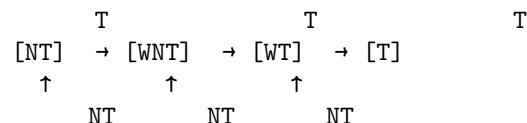
计算给定程序的预测错误次数。

预测器类型

1位预测器状态图:



2位预测器状态图:



经典例题 (2025年6月9日)

```
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];
}
```

分析: – for循环翻译为do-while, 末尾有一个分支 – if语句根据data[i]<min决定跳转

数据	0	1	-2	-3	4	-5	6	-7	-8	9
分支	N	T	N	N	T	N	T	N	N	T

1位预测器(初始NT): – if分支错误: 7次 – for循环: 2次 (开始1次, 结束1次) – 总计: 9次

2位预测器(初始WT): – if分支错误: 6次 – for循环: 1次 (结束时) – 总计: 7次

Cache缓存计算

非考点提示: 根据2026年1月考试说明, Cache相关内容本次不考, 可战略性跳过此章节。

题型

给定地址, 计算tag、set、block-offset, 判断cache hit/miss。

地址分解

: | tag | set index | block offset |
 t s b

- **block offset:** $\log_2(\text{块大小字节数})$
- **set index:** $\log_2(\text{组数})$
- **tag:** 剩余位

2025年6月19日考题

条件: 32位处理器, 4096字节cache, 直接映射, 行长4字节 (16字节)

计算: – 块大小: 4字节 = 16字节 → block offset = 4位 – 组数: $4096/16 = 256$ 组 → set index = 8位 – tag: 32 – 4 – 8 = 20位

地址分析示例: | 地址 | 二进制 | tag | set | offset | ——— | ——— | ——— | ——— | 0x0010 | 0000...0001 0000
| 0x00 | 0x01 | 0x0 | | 0x0014 | 0000...0001 0100 | 0x00 | 0x01 | 0x4 | | 0x0138 | 0000...0011 1000 | 0x00
| 0x13 | 0x8 |

Cache hit判断: 同一set且tag相同则hit

考前速查表

有符号数公式

$$n = -b(n-1) \times 2^{(n-1)} +$$

常见逻辑等价

$$A \cdot C = A \cdot C + \bar{A} \cdot C = (A+C) \cdot (\bar{A}+C)$$

$$A \cdot C = A \cdot C + \bar{A} \cdot C \text{ (XNOR)}$$

香农展开

$$f(Q) = Q \cdot f(0) + \bar{Q} \cdot f(1)$$

RS锁存器

- NOR: S=1置位, R=1复位, 高电平有效
- NAND: S=0置位, R=0复位, 低电平有效

VHDL关键语法

```
rising_edge(clk)      --
(others => '0')      -- 0
&
```

祝你考试顺利!

考前提醒: 1. 提前到达考场 2. 带好笔和计算器 3. 先做会的题 4. 检查答案

最后更新: 2026年1月12日