

LSP 考前突击手册 V2.0

高密度实战版 | 考试: 2026-01-13 10:00 | KN-A-310

1. 考试策略速览

考试范围确认 (来自老师邮件)

必考内容: 有符号/无符号数、K-Map 化简、RS 锁存器、香农展开、流水线

不考内容: Cache 计算、分支预测器

1.1 题型优先级矩阵

题型	分值	难度	优先级
有符号/无符号转换	10-15	低	[S+] 必拿
K-Map 4 变量	12-18	中	[S] 必拿
RS 锁存器状态分析	8-12	中	[A] 高分
香农展开推导	10-15	中高	[A] 高分
流水线 Hazard 分析	15-20	中高	[A] 高分

1.2 时间分配建议 (90 分钟)

时间分配策略

- 数值系统题 (15 分钟): 快速完成, 不要犯低级错误
- K-Map 题 (20 分钟): 画图仔细, 检查圈是否正确
- RS 锁存器/香农 (25 分钟): 按步骤推导, 写清楚过程
- 流水线题 (25 分钟): 画时序图, 标注 Hazard
- 检查 (5 分钟): 检查数值计算和符号

[!] 考场常见失分点

- 有符号数忘记考虑符号扩展
- K-Map 圈的大小不是 2 的幂次
- RS 锁存器 $S=R=1$ 的不稳定状态处理
- 香农展开时代入 0/1 方向搞反
- 流水线 Forwarding 路径画错

2. 数值系统实战

2.1 核心概念速记

n 位数值范围

无符号: 0 到 $2^n - 1$
有符号 (补码): -2^{n-1} 到 $2^{n-1} - 1$

常用值:

- 8 位无符号: 0 到 255
- 8 位有符号: -128 到 +127
- 16 位无符号: 0 到 65535
- 16 位有符号: -32768 到 +32767

2.2 补码转换 - 手把手

负数转补码 (Pro Way)

目标: 将 $-X$ 转为 n 位补码

方法: $-X$ 的补码 = $2^n - X$

例: 8 位表示 -5

- 计算: $2^8 - 5 = 256 - 5 = 251$
- 转二进制: $251 = 128 + 64 + 32 + 16 + 8 + 2 + 1$
- 答案: 11111011

验证: 用传统方法检验

- 5 的二进制: 00000101
- 取反: 11111010
- 加 1: 11111011 ✓

补码转十进制

给定: n 位二进制 $b_{n-1}b_{n-2}...b_1b_0$

有符号解释:

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

例: 11110100 (8 位有符号)

- MSB = 1, 所以是负数
- 计算: $-1 \times 128 + 1 \times 64 + 1 \times 32 + 1 \times 16 + 0 \times 8 + 1 \times 4 + 0 \times 2 + 0 \times 1$
- $= -128 + 64 + 32 + 16 + 4 = -128 + 116 = -12$

2.3 模运算 - 关键考点

模运算核心公式

$$(a+b) \bmod n = ((a \bmod n) + (b \bmod n)) \bmod n$$

$$(a \times b) \bmod n = ((a \bmod n) \times (b \bmod n)) \bmod n$$

$$a \bmod n = a - n \times \lfloor a/n \rfloor$$

负数取模: 结果与除数同号 (Python 风格) 或与被除数同号 (C 风格)

典型例题: 模运算

计算: $(-17) \bmod 5$

方法 1 (加法修正):

- $-17 \div 5 = -3.4$, 取 $\lfloor -3.4 \rfloor = -4$
- $-17 - 5 \times (-4) = -17 + 20 = 3$

方法 2 (直觉法):

- $17 \bmod 5 = 2$
- $-17 \bmod 5 = 5 - 2 = 3$ (使结果为正)

答案: $(-17) \bmod 5 = 3$

2.4 溢出判断

[!] 溢出判断规则

无符号溢出: 进位/借位出最高位

有符号溢出: 两个条件同时满足

- 两个正数相加得负数

- 两个负数相加得正数

判断公式: $V = C_{n-1} \oplus C_n$ (进位异或)

溢出判断实例

计算: 8 位有符号数 $100 + 50$

1. $100 = 01100100$

2. $50 = 00110010$

3. 相加: $01100100 + 00110010 = 10010110$

4. 结果解释: $-128 + 16 + 4 + 2 = -106$

5. 溢出! 两个正数相加得到负数

2.5 符号扩展

符号扩展规则

将 n 位扩展到 m 位 ($m > n$):

无符号: 高位补 0

- 例: 4 位 $1010 \rightarrow$ 8 位 00001010

有符号: 高位复制符号位

- 例: 4 位 $1010 (= -6) \rightarrow$ 8 位 $11111010 (= -6)$

- 例: 4 位 $0110 (= +6) \rightarrow$ 8 位 $00000110 (= +6)$

2.6 快速转换表

十进制	二进制 (4 位)	无符号	有符号
0	0000	0	0
1	0001	1	+1
7	0111	7	+7
8	1000	8	-8
15	1111	15	-1

3. 布尔代数与 K-Map

3.1 布尔代数速查

核心定律

幂等律: $A + A = A, A \cdot A = A$

补余律: $A + \bar{A} = 1, A \cdot \bar{A} = 0$

吸收律: $A + AB = A, A(A + B) = A$

德摩根: $\overline{A + B} = \bar{A}\bar{B}, \overline{AB} = \bar{A} + \bar{B}$

异或: $A \oplus B = A\bar{B} + \bar{A}B$

3.2 卡诺图 - 4 变量标准模板

K-Map 绘制步骤

- 确定变量排列: 行 = AB , 列 = CD (格雷码!)
- 标注格雷码顺序: 00, 01, 11, 10
- 填入真值表中的 1
- 圈相邻的 1 (大小必须是 2 的幂)
- 写出最简 SOP

4 变量 K-Map 模板 (格雷码标注)

		CD			
		00	01	11	10
AB	00	0	1	3	2
	01	4	5	7	6
	11	12	13	15	14
	10	8	9	11	10

3.3 K-Map 圈法规则

[!] K-Map 圈法陷阱

- 圈必须是 2^n 个 1: 1, 2, 4, 8, 16...
- 圈可以跨边界 (环绕!)
- 四角可以组成一个圈!
- 圈越大越好 (消去更多变量)
- 每个 1 至少被圈一次
- 不要重复圈 (除非必要)

K-Map 实战例题

化简: $F(A, B, C, D) = \sum m(0, 1, 2, 5, 7, 8, 9, 10, 14)$

Step 1: 填入 K-Map

		00	01	11	10
00		1	1	0	1
01		0	1	1	0
11		0	0	0	1
10		1	1	0	1

Step 2: 识别圈

- 圈 1 (四角): $m(0, 2, 8, 10) \rightarrow \bar{B}\bar{D}$
- 圈 2 (跨边): $m(0, 1, 8, 9) \rightarrow \bar{B}\bar{C}$
- 圈 3: $m(5, 7) \rightarrow \bar{A}BD$
- 圈 4: $m(14) \rightarrow AB\bar{C}D$

Step 3: 写出最简式

$$F = \bar{B}\bar{D} + \bar{B}\bar{C} + \bar{A}BD + AB\bar{C}D$$

3.4 POS 形式化简

POS 化简步骤

1. 在 K-Map 中圈 0 (不是圈 1!)
 2. 对每个圈写出 maxterm
 3. maxterm 是变量的 OR
 4. 结果是所有 maxterm 的 AND
- 技巧: 圈 0 时, 变量取值与 SOP 相反:
- 变量 =0 → 写变量本身
 - 变量 =1 → 写变量的补

3.5 Don't Care 条件

Don't Care (X) 使用

- Don't Care 可以当 1 用来扩大圈
- Don't Care 也可以当 0 用来忽略
- 目标: 让圈尽可能大
- 最终表达式中不包含 X

4. RS 锁存器与香农展开

4.1 RS 锁存器基础

RS 锁存器真值表

S	R	Q(next)	状态
0	0	Q	保持
0	1	0	复位
1	0	1	置位
1	1	?	禁止/不稳定

特征方程: $Q_{next} = S + \bar{R}Q$ (约束: $SR = 0$)

[!] S=R=1 的问题

- 当 $S = R = 1$ 时:
- NOR 型锁存器: $Q = \bar{Q} = 0$ (矛盾!)
 - NAND 型锁存器: $Q = \bar{Q} = 1$ (矛盾!)
 - 释放后状态不确定 (竞争条件)
- 考试中: 通常标记为“禁止”或“不稳定”

4.2 NOR 型 vs NAND 型

两种 RS 锁存器对比

NOR 型 RS 锁存器:

- $Q = \overline{R + \bar{Q}}$
- $\bar{Q} = \overline{S + Q}$
- 高电平有效 ($S=1$ 置位, $R=1$ 复位)
- 禁止态: $S=R=1$

NAND 型 RS 锁存器:

- $Q = \bar{S} \cdot \bar{\bar{Q}}$
- 低电平有效 ($\bar{S}=0$ 置位, $\bar{R}=0$ 复位)
- 禁止态: $\bar{S}=\bar{R}=0$

4.3 RS 锁存器状态分析

典型例题: 状态序列分析

给定: NOR 型 RS 锁存器, 初始 $Q = 0$

输入序列: $(S, R) = (1, 0) \rightarrow (0, 0) \rightarrow (0, 1) \rightarrow (0, 0) \rightarrow (1, 1)$

Step-by-Step 分析:

时刻	S	R	Q	说明
t_0	-	-	0	初始状态
t_1	1	0	1	置位: $Q \rightarrow 1$
t_2	0	0	1	保持: Q 不变
t_3	0	1	0	复位: $Q \rightarrow 0$
t_4	0	0	0	保持: Q 不变
t_5	1	1	?	禁止态

4.4 香农展开 - 核心考点

香农展开定理

任意布尔函数 $F(x_1, x_2, \dots, x_n)$ 可以对变量 x_i 展开:

$$F = x_i \cdot F_{x_i=1} + \bar{x}_i \cdot F_{x_i=0}$$

其中:

- $F_{x_i=1}$ = 将 x_i 代入 1 后的子函数 (正因子)
- $F_{x_i=0}$ = 将 x_i 代入 0 后的子函数 (负因子)

香农展开步骤 (手把手)

目标: 对 $F(A, B, C)$ 关于变量 A 进行香农展开

步骤:

1. 计算正因子: 令 $A = 1$, 得到 $F_1 = F|_{A=1}$
2. 计算负因子: 令 $A = 0$, 得到 $F_0 = F|_{A=0}$
3. 组合结果: $F = A \cdot F_1 + \bar{A} \cdot F_0$

香农展开完整例题

题目: 对 $F = AB + \bar{A}C + BC$ 关于 A 进行香农展开

Step 1: 计算正因子 F_1 (令 $A = 1$)

$$\begin{aligned} F_1 &= F|_{A=1} \\ &= (1)B + (\overline{1})C + BC \\ &= B + 0 \cdot C + BC \\ &= B + BC \\ &= B \quad (\text{吸收律}) \end{aligned}$$

Step 2: 计算负因子 F_0 (令 $A = 0$)

$$\begin{aligned} F_0 &= F|_{A=0} \\ &= (0)B + (\overline{0})C + BC \\ &= 0 + 1 \cdot C + BC \\ &= C + BC \\ &= C \quad (\text{吸收律}) \end{aligned}$$

Step 3: 组合香农展开式

$$F = A \cdot B + \bar{A} \cdot C$$

验证: 展开结果 $AB + \bar{A}C$

原式 $AB + \bar{A}C + BC$ 可化简:

$$\begin{aligned} &= AB + \bar{A}C + BC(A + \bar{A}) \\ &= AB + \bar{A}C + ABC + \bar{A}BC \\ &= AB(1 + C) + \bar{A}C(1 + B) \\ &= AB + \bar{A}C \quad \checkmark \end{aligned}$$

4.5 多变量香农展开

多变量展开

对两个变量 A, B 同时展开:

$$F = AB \cdot F_{11} + A\bar{B} \cdot F_{10} + \bar{A}B \cdot F_{01} + \bar{A}\bar{B} \cdot F_{00}$$

其中 $F_{ij} = F|_{A=i, B=j}$

两变量香农展开

题目: 对 $F = ABC + \bar{A}\bar{B}C + \bar{A}C$ 关于 A, B 展开
计算四个因子:

- $F_{11} = F|_{A=1, B=1} = (1)(1)C + 0 + (1)\bar{C} = C + \bar{C} = 1$
- $F_{10} = F|_{A=1, B=0} = 0 + 0 + \bar{C} = \bar{C}$
- $F_{01} = F|_{A=0, B=1} = 0 + 0 + 0 = 0$
- $F_{00} = F|_{A=0, B=0} = 0 + (1)(1)C + 0 = C$

结果:

$$F = AB \cdot 1 + A\bar{B} \cdot \bar{C} + \bar{A}B \cdot 0 + \bar{A}\bar{B} \cdot C$$

$$F = AB + A\bar{B}\bar{C} + \bar{A}\bar{B}C$$

4.6 香农展开与 MUX 实现

用 MUX 实现布尔函数

香农展开直接对应 2:1 MUX:

- 选择信号 = 展开变量
- I_0 输入 = 负因子 F_0
- I_1 输入 = 正因子 F_1

级联: 对每个因子继续展开 \rightarrow 多级 MUX 树

[!] 香农展开常见错误

- 代入时忘记处理 \bar{A} (应变成 1 或 0)
- 化简时漏掉吸收律
- MUX 输入接反 (F_0 和 F_1 位置)
- 多变量展开时因子数量算错 (2^n 个)

5. 处理器流水线

5.1 流水线基础

流水线阶段 (5 级经典)

阶段	全称	功能
IF	Instruction Fetch	取指令
ID	Instruction Decode	译码/读寄存器
EX	Execute	执行/ALU 计算
MEM	Memory Access	访存
WB	Write Back	写回寄存器

5.2 流水线时序图

流水线执行时序

指令序列: I1, I2, I3, I4, I5

	C1	C2	C3	C4	C5	C6	C7	C8	C9
I1	IF	ID	EX	MEM	WB				
I2		IF	ID	EX	MEM	WB			
I3			IF	ID	EX	MEM	WB		
I4				IF	ID	EX	MEM	WB	
I5					IF	ID	EX	MEM	WB

性能: 5 条指令用 9 个周期 (理想情况)

5.3 流水线 Hazard 分类

三类 Hazard

1. 结构冲突 (Structural Hazard):

- 硬件资源不足 (如单端口内存)
- 解决: 增加硬件资源

2. 数据冲突 (Data Hazard):

- RAW (Read After Write) - 最常见
- WAW (Write After Write)
- WAR (Write After Read)

3. 控制冲突 (Control Hazard):

- 分支指令导致
- 解决: 预测/延迟槽/Stall

5.4 数据冲突详解

RAW 冲突检测

条件: 指令 I_j 读取指令 I_i 要写的寄存器, 且 I_i 还没写回

例:

I1: ADD R1, R2, R3 ; 写 R1

I2: SUB R4, R1, R5 ; 读 R1 \leftarrow RAW!

时序分析:

- I1 在 C5(WB) 才写回 R1
- I2 在 C3(ID) 就要读 R1

- 冲突: 读发生在写之前!

5.5 Forwarding (旁路)

Forwarding 路径

EX/MEM → EX:

- 将 ALU 结果直接送到下一条指令的 ALU 输入
- 解决 1 周期 RAW 冲突

MEM/WB → EX:

- 将访存结果或旧 ALU 结果送到 ALU 输入
- 解决 2 周期 RAW 冲突

Forwarding 实例

指令:

I1: ADD R1, R2, R3

I2: SUB R4, R1, R5

I3: AND R6, R1, R7

时序 (有 Forwarding):

	C1	C2	C3	C4	C5
I1	IF	ID	EX	MEM	WB
I2		IF	ID	EX*	MEM
I3			IF	ID	EX**

- *EX: 从 I1 的 EX/MEM 寄存器 forward
 - **EX: 从 I1 的 MEM/WB 寄存器 forward
- 无需 Stall!

5.6 Load-Use Hazard

[!] Load 后立即使用

问题: LW 指令在 MEM 阶段才得到数据

I1: LW R1, 0(R2) ; MEM阶段才有R1

I2: ADD R3, R1, R4 ; EX阶段需要R1

即使有 Forwarding 也必须 Stall 1 周期!

	C1	C2	C3	C4	C5	C6
LW	IF	ID	EX	MEM	WB	
ADD		IF	ID	--	EX	MEM

"--" 表示插入的 Stall (气泡)

5.7 Stall 与性能计算

CPI 计算

$$CPI = CPI_{ideal} + \text{Stall cycles per instruction}$$

理想 CPI = 1 (每周期完成一条指令)

实际 CPI = 1 + (数据 Stall + 控制 Stall)

性能计算例题

给定:

- 30% 的指令是 Load
- 50% 的 Load 后面紧跟使用其结果的指令
- 每次 Load-Use 造成 1 周期 Stall

计算 CPI:

$$\text{Stall 率} = 0.30 \times 0.50 = 0.15$$

$$CPI = 1 + 0.15 \times 1 = 1.15$$

5.8 流水线性能公式

吞吐量与加速比

吞吐量:

$$\text{Throughput} = \frac{n \text{ instructions}}{(k + n - 1) \text{ cycles}}$$

其中 k = 流水线级数, n = 指令数

加速比 (vs 非流水线):

$$\text{Speedup} = \frac{n \times k}{k + n - 1}$$

当 $n \rightarrow \infty$: Speedup $\rightarrow k$

5.9 分支处理策略

分支处理方法

1. Stall (最简单):

- 等到分支结果确定再取下一条
- 代价: 每个分支浪费若干周期

2. 预测不跳转:

- 总是取顺序下一条
- 猜错时 Flush 流水线

3. 延迟槽:

- 分支后的指令总是执行
- 编译器负责填充有用指令

考试重点

- 会画流水线时序图
- 能识别 RAW/WAW/WAR 冲突
- 理解 Forwarding 路径
- 知道 Load-Use 必须 Stall
- 计算 CPI 和加速比

老师确认: 会考流水线, 不考分支预测器细节

6. 速查表

6.1 数值系统

概念	公式/规则
n 位无符号范围	$[0, 2^n - 1]$
n 位有符号范围	$[-2^{n-1}, 2^{n-1} - 1]$
负数补码	$-X \rightarrow 2^n - X$
补码转值	$-b_{n-1} \cdot 2^{n-1} + \sum b_i \cdot 2^i$
符号扩展	复制 MSB
溢出条件	正 + 正 = 负或负 + 负 = 正

6.2 卡诺图

规则	说明
格雷码顺序	00, 01, 11, 10
圈大小	必须是 2^n (1,2,4,8...)
环绕	左右/上下边可连
四角	可组成一个圈
Don't Care	可当 1 扩大圈

4 变量索引:

	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

6.3 RS 锁存器

S	R	Q_{next}	说明
0	0	Q	保持
0	1	0	复位
1	0	1	置位
1	1	?	禁止

特征方程: $Q_{next} = S + \bar{R}Q$

6.4 香农展开

$$F = x \cdot F_{x=1} + \bar{x} \cdot F_{x=0}$$

步骤:

1. 令 $x = 1$, 计算 F_1
2. 令 $x = 0$, 计算 F_0
3. 组合: $F = xF_1 + \bar{x}F_0$

6.5 流水线

5 级: IF \rightarrow ID \rightarrow EX \rightarrow MEM \rightarrow WB

冲突类型:

- RAW: 读后写 (最常见)
- Forwarding: EX/MEM \rightarrow EX, MEM/WB \rightarrow EX
- Load-Use: 必须 Stall 1 周期

性能:

$$CPI = 1 + \text{Stall 率}$$

$$\text{Speedup} = \frac{nk}{k+n-1} \rightarrow k$$

6.6 布尔代数

德摩根	$\overline{A+B} = \bar{A}\bar{B}, \overline{AB} = \bar{A} + \bar{B}$
吸收律	$A + AB = A, A(A+B) = A$
异或	$A \oplus B = A\bar{B} + \bar{A}B$
补余	$A + \bar{A} = 1, A\bar{A} = 0$

6.7 常用数值

2^n	值	n 位 max	有符号
2^4	16	15	± 7
2^8	256	255	± 127
2^{10}	1024	1023	± 511
2^{16}	65536	65535	± 32767

Good Luck!

2026-01-13 10:00 | KN-A-310