LSP Exam - January 30, 2015

> **CVUT FEL (ČVUT) - České vysoké učení technické v Praze | Czech Technical University in Prague**
>
> [ ](2015-01-30_Exam_CN.md) | [English](2015-01-30_Exam_EN.md) | [Čeština](2015-01-30_Exam_CZ.m

> **AI-Generated Solution** - Reference analysis below

Question 1: Logic Expression Simplification

Problem: Rewrite the logic expression F so that NOT operators only appear before variables, verify

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

Solution Steps

1. Apply De Morgan's law: `not(X or Y) = not X and not Y`
2. Simplify step by step...

```
F = not(A or B or C) and not((A or B) and not(A or C))
  = Ā·B·C and (not(A or B) or (A or C))
  = Ā·B·C and ((Ā·B) or A or C)
```

Continue simplifying to get the final result...

Karnaugh Map Verification

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

Question 2: RS Latch Circuit Simulation Frequently Tested

Problem: Given inputs A, B, C values at times t0, t1, t2, t3 as shown, write the Q output value.

```
A = ..0..|..1..|..1..|..1..|
B = ..0..|..0..|..0..|..1..|
C = ..1..|..1..|..0..|..0..|

t0   t1   t2   t3

Q = _____|_____|_____|_____|
```

```

Assume the intervals between input changes are long enough to ignore gate delays.

---

## Question 3: Shannon Expansion  Frequently Tested

**Problem**: Decompose the function `Q=f(A,B,C,Q)` from question 2 into:

```
Q = (not Q and f0(A,B,C)) or (Q and f1(A,B,C))
```

Write the Karnaugh maps for f0 and f1.

### Solution Method
1. Set Q=0, find f0
2. Set Q=1, find f1
3. Draw Karnaugh maps

---

## Question 4: Signed/Unsigned Bit Values  Frequently Tested

**Problem**: What is the decimal value of the 10-bit binary number `10 0000 1111`?

a) **Unsigned**: _____

b) **Two's-complement (signed)**: _____

### Answer

```
10 0000 1111 (binary)

a) Unsigned:
   = 2^9 + 2^3 + 2^2 + 2^1 + 2^0
   = 512 + 8 + 4 + 2 + 1
   = 527

b) Two's-complement (signed):
   MSB is 1, indicating negative number
   Method 1: = -2^9 + (remaining bits value)
           = -512 + 15 = -497

   Method 2: Invert and add 1
           01 1111 0000 + 1 = 01 1111 0001 = 497
           So the original number = -497
```

Question 5: Equivalent Logic Functions Frequently Tested
```

Problem: Mark all logic functions that are equivalent to other functions:

```vhdl
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);
```

Solution Method
Draw Karnaugh maps for each function:

f1: (A xor C) or (A and not C) = A or (A xor C) = A or C·C...

f4: (A or C) and (not A or not C) = A C

f6: (A and not C) or (C and not A) = A C

Conclusion: f4 f6 (both are A XOR C)

Question 6: RS Latch Drawing Frequently Tested

Problem: Draw RS latch using only NOR gates, and draw RS latch using only NAND gates.

NOR-type RS Latch (active-high)
```
S   >    NOR          Q




R   >    NOR          Q


```

NAND-type RS Latch (active-low)
```
S   >   NAND          Q
```

```
R   >   NAND          Q


```

Question 7: Asynchronous Divide-by-18 Divider Design

Problem: Add gates and connections to the incomplete diagram to create an asynchronous divide-by-18

Design Approach
- Divide-by-18 requires counting 0-17, total 18 states
- 17 in binary: 10001
- Reset all flip-flops when 17 is detected

Key Connections
- Use 5 D flip-flops in cascade
- Use AND gate to detect state 10001
- Connect detection output to CLRN of all DFFs

Question 8: VHDL Code Analysis

Problem: Analyze the poorly formatted VHDL code and draw the corresponding logic circuit diagram.

```vhdl
library IEEE; use IEEE.STD_LOGIC_1164.all;
entity test20140214 is port (a, b, c, d : in std_logic; e : out std_logic); end;
architecture rtl of test20140214 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;
```

Formatted Code
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity test20140214 is
    port (a, b, c, d : in std_logic;
```

```
          e : out std_logic);
end;

architecture rtl of test20140214 is
begin
    process(a, b)
        variable z: std_logic_vector(0 to 3);
    begin
        if b = '0' then
            z := (others => '0');     -- Async clear
        elsif rising_edge(a) then
            if c = '1' then
                z := d & z(0 to 2);   -- Serial input d
            else
                z := z(3) & z(0 to 2); -- Circular shift
            end if;
        end if;
        e <= z(3);
    end process;
end rtl;
```

Functional Analysis
- **Circuit Name**: 4-bit controllable shift register
- **a**: Clock signal (rising edge triggered)
- **b**: Async clear (clears when b='0')
- **c**: Mode control
  - c='1': Serial input mode, d enters z(0)
  - c='0': Circular shift mode
- **e**: Output z(3)

Knowledge Summary

Question	Type	Difficulty
1	Logic expression simplification	
2	RS latch simulation	
3	Shannon expansion	
4	Signed/Unsigned numbers	
5	Equivalent function identification	
6	RS latch drawing	
7	Divider design	
8	VHDL analysis	