



Försättsblad Prov Original

Kurskod	Provkod	Tentamensdatum
E T O 6 1 G	T 1 0 2	2 0 1 8 - 0 3 - 2 3
Kursnamn	Elektroteknik GR (A), Digitalteknik med VHDL	
Provnamn	Skriftlig tentamen	
Ort	Sundsvall	
Termin	V18	
Ämne	Elektroteknik	

MID SWEDEN UNIVERSITY, SWEDEN

Examination – ET061G– 23 March 2018

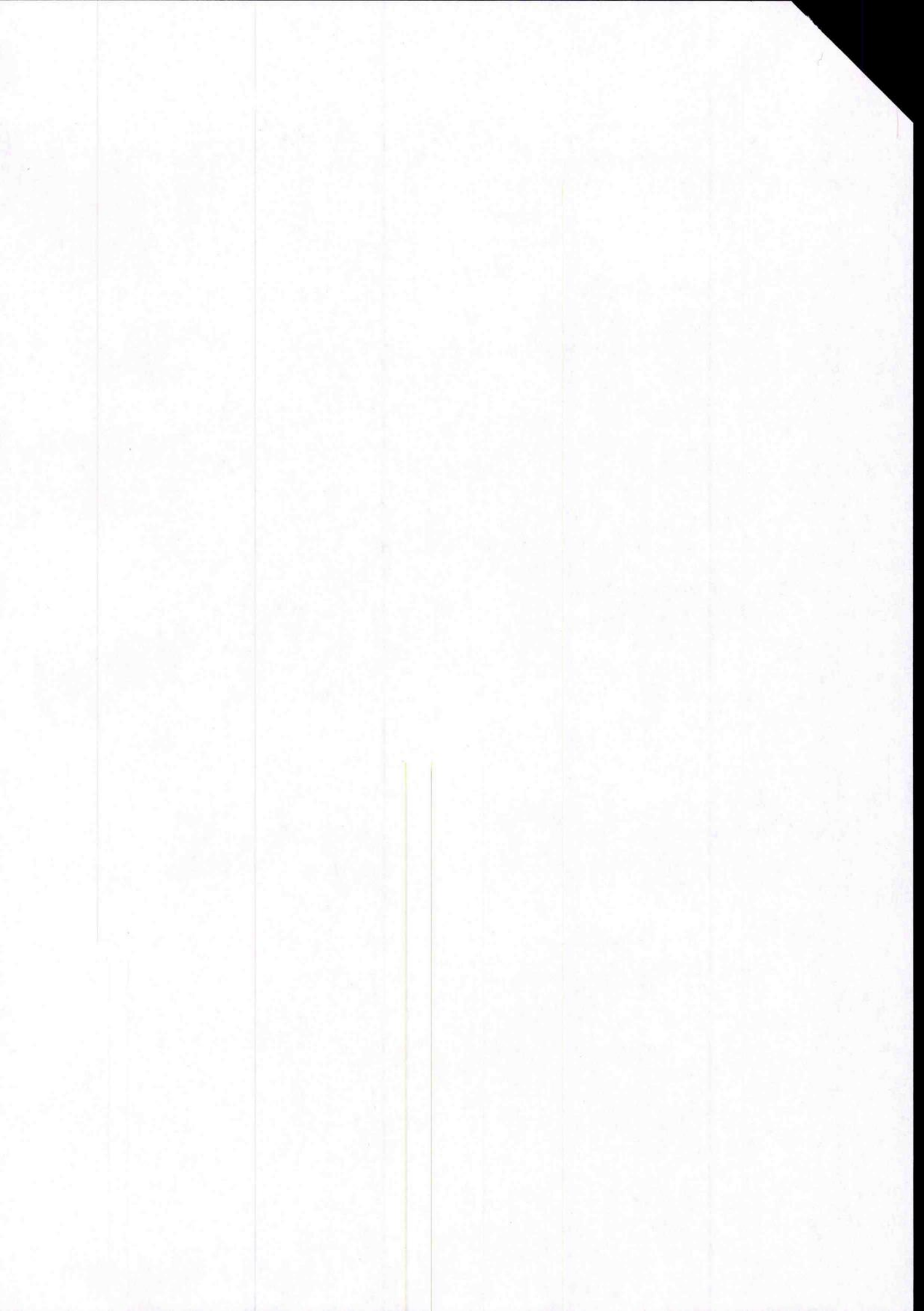
Digital Electronics with VHDL

Time:	Five hours
Permitted help material:	None
Total number of questions:	4
Total number of pages/sidor:	5
Max point:	55 (25 points are required to pass)

Najeem Lawal, Tel: 010-142 8561, E-post: najeem.lawal@miun.se

Instructions:

- Please provide the solutions, explanations and arguments in English.
- Please submit a question per sheet.
- Reasons and motivations must not be so few that it will be difficult to follow.
- The thinking behind equations and axioms set must be explained.
- The deductions must be complete to clearly show how results have been obtained.
- Each solution must be completed with a clearly marked answer.



Useful axioms for Boolean algebra:

- | | | |
|--------------------|-------------------------------|----------------------------------|
| 1. $A + 0 = A$ | 5. $A + \overline{A} = A$ | 9. $\overline{\overline{A}} = A$ |
| 2. $A + 1 = 1$ | 6. $A + \overline{A} = 1$ | 10. $A + AB = A$ |
| 3. $A \cdot 0 = 0$ | 7. $A \cdot A = A$ | 11. $A + \overline{A}B = A + B$ |
| 4. $A \cdot 1 = A$ | 8. $A \cdot \overline{A} = 0$ | 12. $(A + B)(A + C) = A + BC$ |

Questions

Questions 1.

10 P.

- Convert **11000111** from binary to decimal number (2 P.).
- Multiply the binary number in **1a)** by 4 by using the left-shift operator and write the result as a decimal number (2 P.).
- Divide the binary number in **1a)** by 2 by using the right-shift operator and write the result as a decimal number (2 P.).
- Convert the hexadecimal number **DB3** into binary. Then convert the obtained binary number into octal (2 P.).
- Calculate **-5+3** by using 2's complement (2 P.).

Questions 2.

15 P.

- Determine if the following equality is true using Boolean Algebra (5 P.):

$$a \cdot \overline{b} + (c + \overline{d}) = \overline{\overline{a \cdot \overline{b} \cdot c \cdot \overline{a} \cdot \overline{b} \cdot \overline{d}}}$$

- Minimize the following Boolean expression (5 P.):

$$(x + y + z)(x + y + \overline{z})$$

- Minimize the following function by using a Karnaugh-diagram.

$$f(x, y, z, w) = \sum(1, 2, 3, 4, 6, 7, 9, 11) + d(0, 5, 10)$$

Write the result as a minimal Sum-of-Products (SoP) expression and draw the circuit diagram for the minimized expression (5 P.).

Questions 3.

15P.

- a) If the following VHDL code snippet is correct, what is the data type of signal **sel**? (2P.)

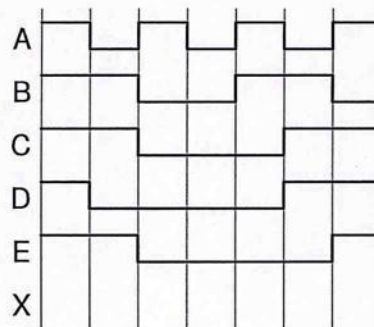
```

case sel is
  when '0' => z <= a;
  when '1' => z <= b;
end case;
    
```

Re-write the code snippet to avoid latches using datatype **std_logic** for signal **sel** using **if-then** statements. (3P.)

- b) Write a function in VHDL that takes a 5-bit input vector (1 bit for odd parity + 4 bit data), and then verifies if the parity bit matches the data or not (i.e. checks if the data has been correctly transmitted). (5P.)
- c) Draw the circuit diagram for the expression below and complete the signal for the output X (5P.).

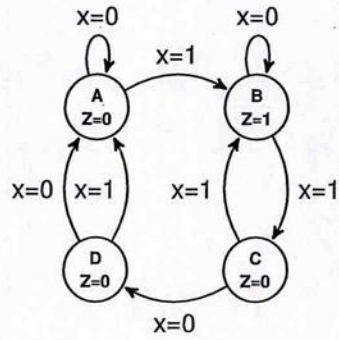
$$X = AC + \overline{A}B + ED + C + \overline{B}\overline{E}$$



Questions 4.

15P.

- a) Write a complete synthesizable VHDL code to implement the state machine in the diagram below. The code must have three different processes. The processes should include combinatorial process for state transition, clocked process for state transfer and process for output estimation. Expected input signal are **clk** and **x** whereas the output is **Z**. Use Appendix 1 as inspiration. (6P.)



- b) Develop Boolean expressions for the state transition graph in Question 4a. Use gray coding and D-flipflops (6 P.).
- c) Draw the state machine diagram for the VHDL code in Appendix 1. Is this a Moore or Mealy state machine? Motivate your answer (3 P.)

Appendix 1

```
library ieee;
use ieee.std_logic_1164.all;
ENTITY q4c IS
PORT (
    clk, reset, in1    : IN  STD_LOGIC;
    out1                : OUT STD_LOGIC_VECTOR(1 DOWNTO 0) );
END q4c;
ARCHITECTURE fsm OF q4c IS
    TYPE state_type IS (s0, s1, s2, s3); -- state declaration
    SIGNAL current_state, next_state : state_type;
BEGIN
    p0 : PROCESS(current_state, in1) --state transistion combinational process
    BEGIN
        CASE current_state IS
            WHEN s0 => IF in1 = '1' THEN
                next_state<=s1;
            END IF;
            WHEN s1 => IF in1 = '0' THEN
                next_state<=s3;
            END IF;
            WHEN s2 => IF in1 = '0' THEN
                next_state<=s0;
            END IF;
            WHEN s3 => IF in1 = '1' THEN
                next_state<=s2;
            END IF;
            WHEN OTHERS => null;
        END CASE;
    END PROCESS p0;

    p1 : PROCESS(clk, reset) --state transfer clocked process
    BEGIN
        IF reset = '1' THEN
            current_state <= s0;
        ELSIF clk'EVENT AND clk='1' THEN
            current_state <= next_state;
        END IF;
    END PROCESS p1;

    p3 : PROCESS (current_state) --output estimation combinational process
    BEGIN
        CASE current_state IS
            WHEN s0 => out1 <= "00";
            WHEN s1 => out1 <= "01";
            WHEN s2 => out1 <= "11";
            WHEN s3 => out1 <= "10";
            WHEN OTHERS => null;
        END CASE;
    END PROCESS p3;
END fsm;
```