Group - 28 Assignment-2

24CS10015 : Jakkala Naga Vamsi Krishna
24CS10018 : Miryala Sai Nikhil Chakravarthi

A) Design and Implementation of a 4-bit Prime Number Detector

Aim

To design and implement a combinational logic circuit that accepts a 4-bit binary number

$$N = (A, B, C, D)$$

where A is the Most Significant Bit (MSB) and D is the Least Significant Bit (LSB), and produces an output

$$F = 1$$

if the number is a **prime number**, and

$$F = 0$$

otherwise. The circuit is implemented using TTL logic ICs and verified using an external LED.

Apparatus

- IC 74LS04 – Hex NOT gate
- IC 74LS11 – Triple 3-input AND gate (2 ICs)
- IC 74LS32 – Quad 2-input OR gate
- Breadboard
- LED
- 680 Ω resistor
- 5 V regulated DC power supply
- Connecting wires

Theory

A 4-bit binary number can represent decimal values from 0 to 15. The prime numbers in this range are:

$$2,\ 3,\ 5,\ 7,\ 11,\ 13$$

The output must be HIGH only for these values.

Truth Table

Decimal	A	B	C	D	F
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	0

Minterm Representation

From the truth table,

$$F(A, B, C, D) = \Sigma m(2, 3, 5, 7, 11, 13)$$

Law used: Definition of minterms.

Canonical SOP Expansion

$$\begin{aligned} F = {} & \bar{A}\bar{B}C\bar{D} + \bar{A}\bar{B}CD + \bar{A}B\bar{C}D + \bar{A}BCD \\ & + A\bar{B}CD + AB\bar{C}D \end{aligned}$$

Expression Simplification

The canonical SOP expression is simplified using Boolean algebra to obtain a reduced form suitable for hardware implementation using available TTL ICs.

The final simplified Boolean expression used for circuit realization is:

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

This expression produces output $F = 1$ only for prime numbers between 0 and 15 and $F = 0$ for all other input combinations.

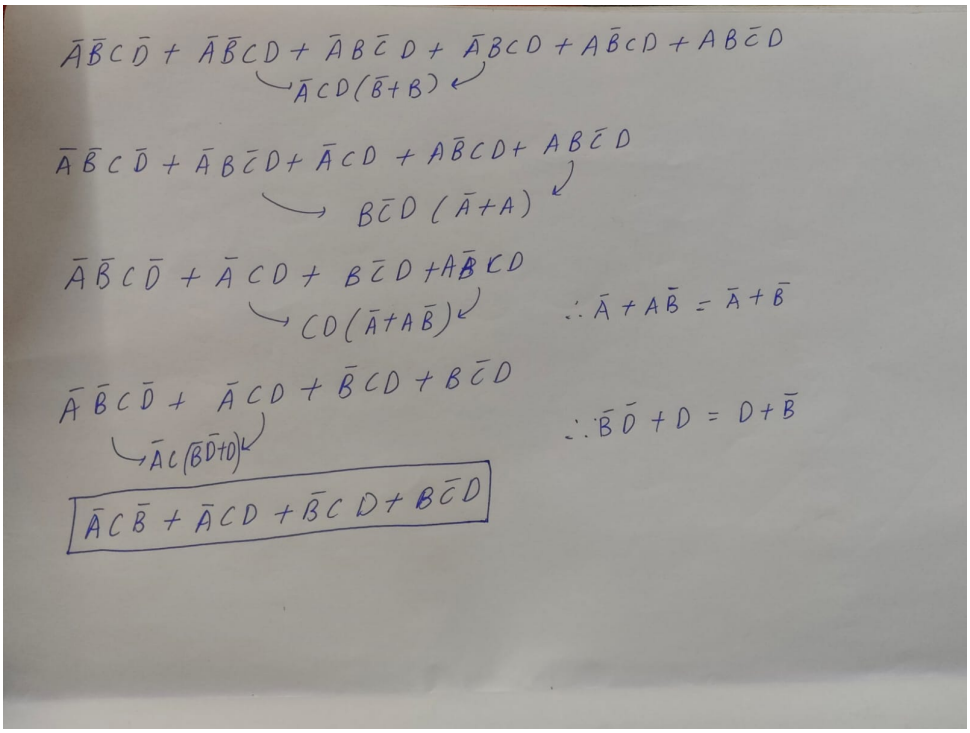

Figure 1: simplification

IC Selection Justification

- IC 74LS04 is used to generate the complemented inputs $\bar{A}$, $\bar{B}$, and $\bar{C}$
- D is not inverted since $\bar{D}$ is not required
- Each product term contains three variables, hence 3-input AND gates are used
- One 74LS11 provides only three AND gates, so two ICs are required
- OR operation is implemented using IC 74LS32

Circuit Diagram

The logic circuit is implemented based on the simplified Boolean expression:

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

- IC 74LS04 is used to generate inverted signals
- IC 74LS11 is used to implement the four 3-input AND gates
- IC 74LS32 is used to OR the outputs of the AND gates

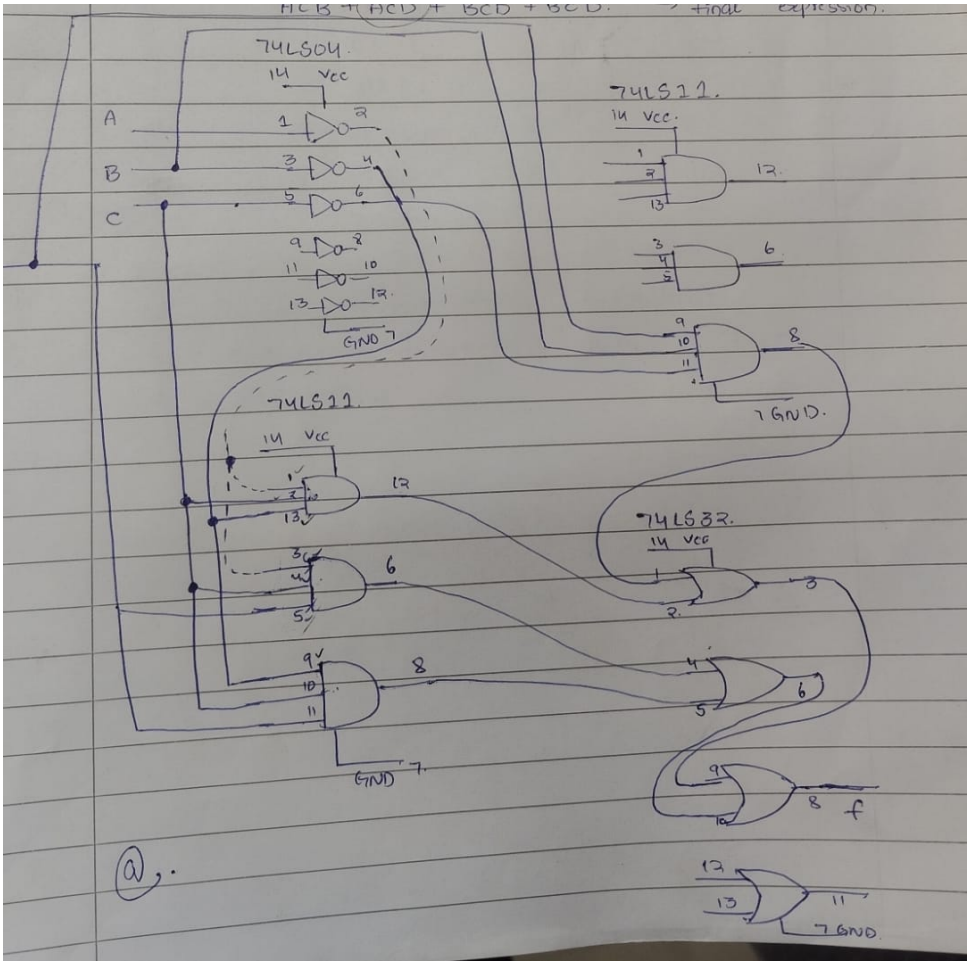

Figure 2: Logic circuit for 4-bit prime number detector

- The output is connected to an external LED through a current-limiting resistor

The complete circuit diagram with IC pin numbers and power connections is drawn neatly in the laboratory record.

Power Supply Connections

For all ICs:

$$\text{Pin } 14 \rightarrow +5\,\text{V}, \quad \text{Pin } 7 \rightarrow \text{GND}$$

Pin Connections

74LS04 – NOT Gate

Input	Input Pin	Output Pin
A	1	2 ($\bar{A}$)
B	3	4 ($\bar{B}$)
C	5	6 ($\bar{C}$)
D	Not connected ($\bar{D}$ not required)	

74LS11 – IC 1

Expression	Input Pins	Output Pin
$\bar{A}\bar{B}C$	1, 2, 13	12
$\bar{A}BD$	3, 4, 5	6
$\bar{B}CD$	9, 10, 11	8

74LS11 – IC 2

Expression	Input Pins	Output Pin
$B\bar{C}D$	1, 2, 13	12

OR Gate Implementation (74LS32)

The outputs of the AND gates are combined using IC 74LS32 (Quad 2-input OR gate). The intermediate OR outputs are denoted as OR1 and OR2.

Inputs	Input Pins	Output Pin
AND1, AND2	1, 2	3 (OR1)
AND3, AND4	4, 5	6 (OR2)
OR1, OR2	9, 10	8 (F)

Thus,

$$\text{OR1} = \text{AND1} + \text{AND2}$$

$$\text{OR2} = \text{AND3} + \text{AND4}$$

$$F = \text{OR1} + \text{OR2}$$

External LED Connection

Resistor Calculation

$$R = \frac{V_{CC} - V_{LED}}{I_{LED}} = \frac{5 - 2}{5 \times 10^{-3}} = 600\Omega$$

A standard resistor of

$$\boxed{680\Omega}$$

is used.

Connection

$$F \rightarrow \text{LED anode} \rightarrow 680\Omega \rightarrow \text{GND}$$

Procedure

1. Connect all ICs to the power supply.
2. Apply inputs A, B, C, and D using switches.
3. Verify the output using the external LED.

Observation

The LED glows only when the input binary number corresponds to a prime number.

Result

A 4-bit prime number detector was successfully designed, implemented, and tested using TTL logic ICs.

Conclusion

This experiment demonstrates the application of Boolean algebra and combinational logic design to detect prime numbers using standard TTL logic ICs.

B) Implementing a circuit for B = (A + 4)% mod 10 using only 2-input NAND gates

Aim

To design and implement a combinational circuit that takes a BCD digit

$$A = (A_3, A_2, A_1, A_0)$$

as input and produces a BCD digit

$$B = (B_3, B_2, B_1, B_0)$$

as output, where

$$B = (A + 4) \bmod 10.$$

The circuit is to be realized using **only 2-input NAND gates** and both the input and output digits are to be displayed on two separate 7-segment LED displays.

Apparatus

1. Breadboard
2. IC 7400 (Quad 2-input NAND gates)
3. IC 7447 (BCD to 7-segment decoder driver) – 2 units
4. Common anode 7-segment LED display – 2 units
5. Connecting wires
6. DC power supply (+5V)

Theory

A Binary Coded Decimal (BCD) number represents decimal digits from 0 to 9 using four binary bits. The input BCD digit is given as

$$A_3A_2A_1A_0 \quad (A_3 \text{ is MSB, } A_0 \text{ is LSB})$$

The circuit computes

$$B = (A + 4) \bmod 10$$

Hence, for valid BCD inputs (0–9), the output is also a valid BCD digit. Inputs from 10 to 15 are invalid BCD inputs and are treated as don't-care conditions.

The outputs B_3, B_2, B_1, B_0 are expressed as Boolean functions of the inputs and implemented using NAND gates only.

Calculations

Truth Table

A_3	A_2	A_1	A_0	Decimal A	Decimal B	B_3	B_2	B_1	B_0
0	0	0	0	0	4	0	1	0	0
0	0	0	1	1	5	0	1	0	1
0	0	1	0	2	6	0	1	1	0
0	0	1	1	3	7	0	1	1	1
0	1	0	0	4	8	1	0	0	0
0	1	0	1	5	9	1	0	0	1
0	1	1	0	6	0	1	0	1	0
0	1	1	1	7	1	1	0	1	1
1	0	0	0	8	2	0	0	0	0
1	0	0	1	9	3	0	0	0	1

Inputs 10–15 are invalid BCD inputs and are treated as don't-care conditions.

Logic Expressions

From the truth table, the Boolean expressions are obtained directly as:

$$B_0 = A_0$$

$$B_1 = \overline{A_2}(A_1 \oplus A_3)$$

$$B_2 = \overline{A_2A_3}$$

$$B_3 = A_2\overline{A_1}$$

Realization Using NAND Gates Only

Using the identities:

$$\overline{X} = X \text{ NAND } X$$
$$X + Y = \overline{\overline{X} \cdot \overline{Y}}$$
$$X \cdot Y = \overline{X \text{ NAND } Y}$$

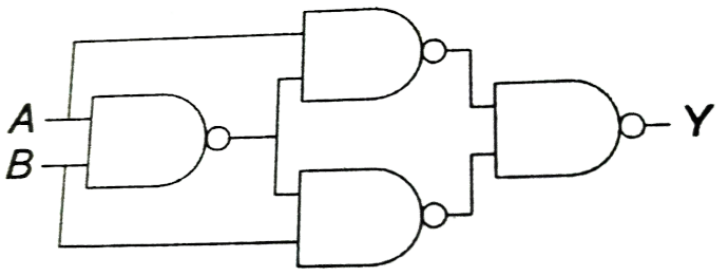

Figure 3: XOR using 2-input NAND

Each output function is rewritten using only NAND operations and implemented using IC 7400. All NOT, AND, and OR operations are realized using combinations of 2-input NAND gates.

Procedure

1. Connect all ICs to the power supply (+5V and ground)
2. Apply BCD input $A_3A_2A_1A_0$
3. Construct the logic for B_3, B_2, B_1, B_0 using NAND gates only
4. Connect the outputs to the second BCD-to-7-segment decoder
5. Observe the input and output digits on the respective 7-segment displays

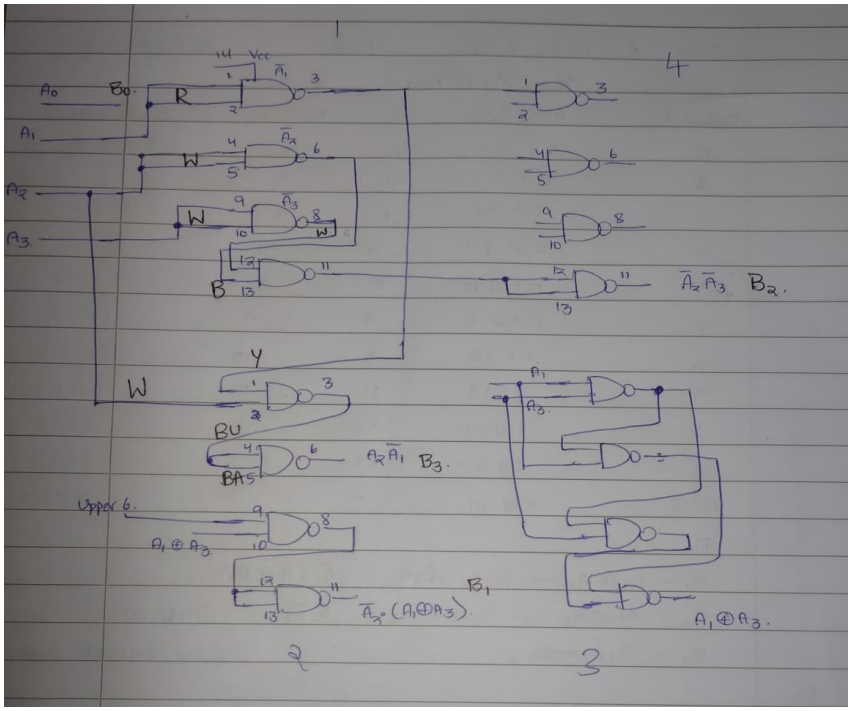

Figure 4: Circuit Diagram

7-Segment Display Connection

Connect the 7-Segment display connection by following the given guidelines

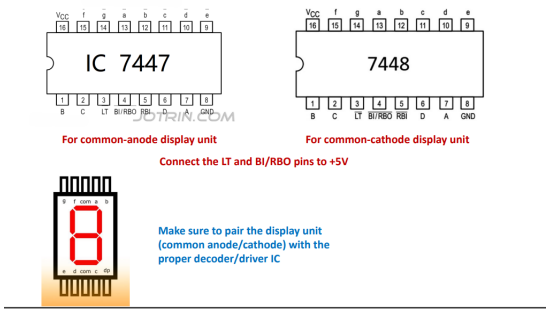

Figure 5: IC 7447 pins

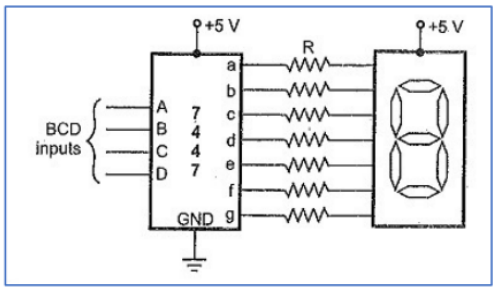

Figure 6: Connecting IC and Display

Observations

- For valid BCD inputs (0–9), the output displayed is equal to $(A + 4) \bmod 10$
- Both input and output digits are correctly displayed on the 7-segment displays
- The circuit behaves correctly for all valid inputs

Result

A combinational circuit was successfully designed and implemented using only 2-input NAND gates to compute

$$B = (A + 4) \bmod 10$$

for a BCD input A. The input and output digits were correctly displayed using two 7-segment LED displays.