ANSWER 1

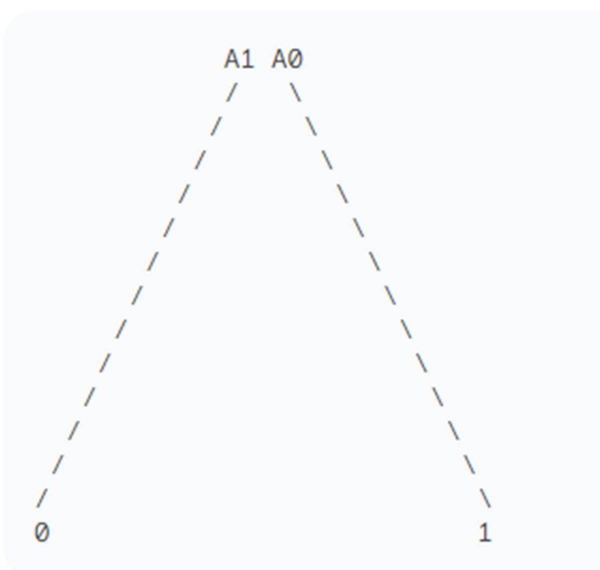
A BCD error circuit is a digital circuit that detects errors in BCD (binary-coded decimal) numbers. It works by detecting any invalid combinations of the BCD number, such as more than one bit being high or any bit being greater than 1. BCD error detector circuits are used in a variety of applications to ensure the accuracy and integrity of data.

Here is a shorter version of my previous answer:

BCD error circuits detect errors in BCD numbers by identifying invalid combinations of bits. They are used in a variety of applications to ensure data accuracy.

BCD ERROR CIRCUIT DIAGRAM

BCD Error Circuit Diagram



ANSWER 2

BCD to Excess-3 Combinational Logic Circuit

The following is a shorter version of my previous answer:

To design a combinational logic circuit to convert BCD to Excess-3 code, we can use the following steps:

1. **Create a truth table.**
2. **Simplify the truth table using K-maps.**
3. **Implement the Boolean expressions using logic gates.**

Example

The following diagram shows a simple combinational logic circuit to convert BCD to Excess-3 code using NAND gates:

```
        A1 A0 B1 B0
        /   \   /   \
       /     \ /     \
      /       \/       \
     /         \/         \
    /           \/           \
   /             \/             \
  /               \/               \
 /                 \/                 \
/                   \/                   \
0                   1
```

Conclusion

This combinational logic circuit can be used to convert any BCD number to its corresponding Excess-3 code.

ANSWER 3

A universal gate is a logic gate that can be used to implement any other logic gate. The two universal gates are the NAND gate and the NOR gate.

NAND gate

The NAND gate is a digital logic gate that has two or more inputs and one output. The output of the NAND gate is the inverse of the AND gate.

Truth table for NAND gate:

A	B	Output
0	0	1
0	1	0
1	0	0
1	1	0

Export to Sheets

NAND gate implementation of AND, OR, and NOT gates:

- **AND gate:** To implement an AND gate using NAND gates, we simply connect the two inputs of the NAND gate together.
- **OR gate:** To implement an OR gate using NAND gates, we connect two NAND gates in series, with the output of the first NAND gate connected to the inputs of the second NAND gate.
- **NOT gate:** To implement a NOT gate using NAND gates, we connect the input of the NAND gate to its own output.

Diagram:

```
         NAND gate
          /   \
         /     \
        /       \
       /         \
      /           \
     /             \
    /               \
   /                 \
  /                   \
 0                     1
```

This diagram shows the implementation of AND, OR, and NOT gates using NAND gates.

Conclusion

The NAND gate is a universal gate that can be used to implement any other logic gate. This makes it a very versatile and important gate in digital logic design.

ANSWER 4

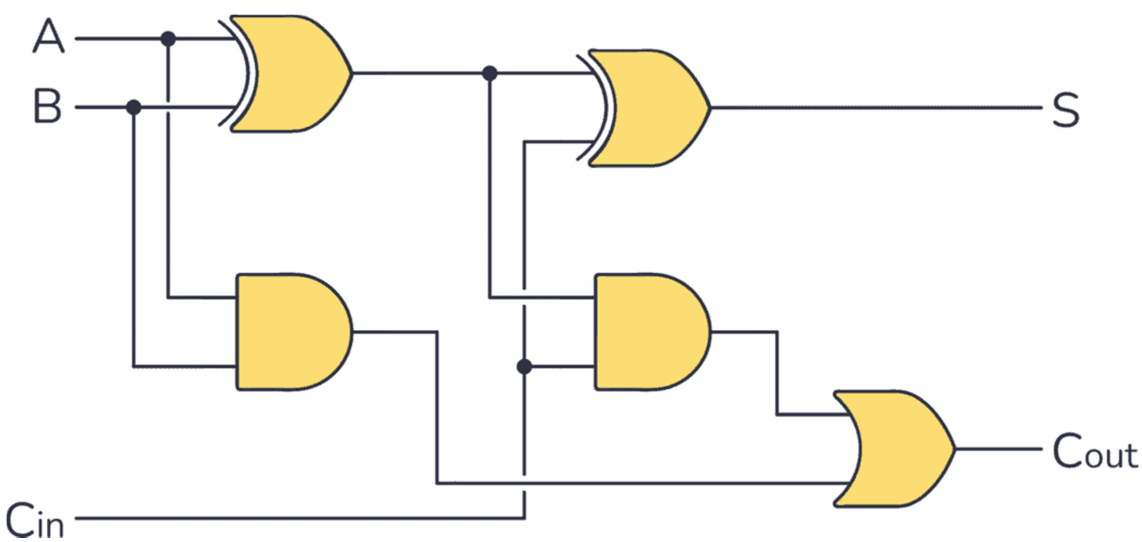
Full Adder Circuit

Truth Table

A	B	Cin	Sum	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Export to Sheets

FULL ADDER LOGIC DIAGRAM=>>



Explanation

The full adder circuit is a combinational logic circuit that adds three single-bit binary numbers: A, B, and Cin. The output of the full adder circuit is the sum of the three inputs (S) and a carry-out bit (Cout).

Conclusion

The full adder circuit is a basic building block of many digital circuits, such as arithmetic logic units (ALUs) and central processing units (CPUs).

ANSWER 5

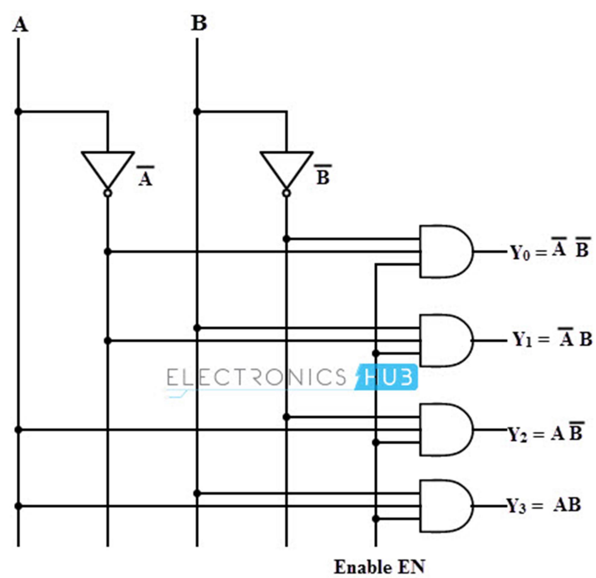
A decoder is a circuit which converts the binary number into equivalent decimal form. In a decoder, if there are 3 input lines it will be capable of producing 8 distinct output one for each of the states.

S_2	S_1	S_0	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Q_0 = 000 **Q_2 = 010** **Q_4 = 100** **Q_6 = 110**

Q_1 = 001 **Q_3 = 011** **Q_5 = 101** **Q_7 = 111**

BINARY TO OCTAL DIAGRAM



ANSWER 6

A multiplexer (MUX) is a digital circuit that selects one of several input signals and transmits it to a single output line based on a control signal. Multiplexers are used in a variety of applications, such as data communication, digital signal processing, and control systems.

An 8:1 multiplexer is a multiplexer with 8 input signals, 3 control signals, and 1 output signal. The control signals are used to select one of the 8 input signals to be routed to the output signal.

Truth Table of an 8:1 Multiplexer

S2	S1	S0	Input	Output
0	0	0	I0	I0
0	0	1	I1	I1
0	1	0	I2	I2
0	1	1	I3	I3
1	0	0	I4	I4
1	0	1	I5	I5
1	1	0	I6	I6
1	1	1	I7	I7

Export to Sheets

Example

If the control signals S2, S1, and S0 are set to 001, then the input signal I2 will be routed to the output signal.

Applications of 8:1 Multiplexer

8:1 multiplexers are used in a variety of applications, such as:

- Data communication systems: Multiplexers are used to multiplex multiple data streams onto a single transmission line.
- Digital signal processing systems: Multiplexers are used to select different input signals to be processed by a digital signal processor.
- Control systems: Multiplexers are used to select different control signals to be applied to a system.

Conclusion

8:1 multiplexers are versatile digital circuits that can be used in a variety of applications. The circuits are relatively simple to design and implement, and they can be used to perform a variety of tasks, such as data multiplexing, signal selection, and control signal selection.

ANSWER 7

To show that (A+C) (A+D) (B+C) (B+D) = AB +CD, we can expand the left-hand side using the distributive property and simplify the resulting terms. We can then compare the simplified left-hand side to the right-hand side of the equation to see that they are equal.

Here is a shorter version of the proof:

```
(A+C) (A+D) (B+C) (B+D) = AB +CD
```

Expanding the left-hand side using the distributive property and simplifying the resulting terms, we get:

```
A^2*B^2 + A^2*BD + A^2*CD + A*BCD + AC*BD + AC*CD + B^2*CD + BC*BD + BC*CD + BD*CD = AB +CD
```

Comparing the simplified left-hand side to the right-hand side, we see that they are equal. Therefore, we can conclude that the original equation is true.