**LAB MANUAL (DIGITAL ELECTRONICS)**

|  |
| --- |
| **EXPERIMENT NO:1**  Verification and interpretation of truth tables for AND, OR, NOT, NAND, NORExclusive OR (EX-OR), Exclusive NOR (EX-NOR) Gates.    **Apparatus:** Logic trainer kit, logic gates / ICs, wires.  **Theory:**  Logic gates are electronic circuits which perform logical functions on one or more inputs to produce one output. There are seven logic gates. When all the input combinations of a logic gate are written in a series and their corrresponding outputs written along them, then this input/ output combination is called **Truth Table**. Various gates and their working is explained here.    **AND Gate**  AND gate produces an output as 1, when all its inputs are 1; otherwise the output is 0. This gate can have minimum 2 inputs but output is always one. Its output is 0 when any input is 0.  [https://sites.google.com/site/amtmttl/_/rsrc/1470268675985/st2/IC7408.PNG](https://sites.google.com/site/amtmttl/st2/IC7408.PNG?attredirects=0)  **IC 7408**  ---------------------------------------------------------------------------------------------------    **OR Gate**  OR gate produces an output as 1, when any or all its inputs are 1; otherwise the output is 0. This gate can have minimum 2 inputs but output is always one. Its output is 0 when all input are 0.  [https://sites.google.com/site/amtmttl/_/rsrc/1470268676307/st2/IC7432.PNG](https://sites.google.com/site/amtmttl/st2/IC7432.PNG?attredirects=0)  IC 7432  ---------------------------------------------------------------------------------------------------  **NOT Gate**  NOT gate produces the complement of its input. This gate is also called an INVERTER. It always has one input and one output. Its output is 0 when input is 1 and output is 1 when input is 0.  [https://sites.google.com/site/amtmttl/_/rsrc/1470268676389/st2/IC7404.PNG](https://sites.google.com/site/amtmttl/st2/IC7404.PNG?attredirects=0)  IC 7404  ---------------------------------------------------------------------------------------------------  **NAND Gate**  NAND gate is actually a series of AND gate with NOT gate. If we connect the output of an AND gate to the input of a NOT gate, this combination will work as NOT-AND or NAND gate. Its output is 1 when any or all inputs are 0, otherwise output is 1.    [https://sites.google.com/site/amtmttl/_/rsrc/1470268676152/st2/IC7400.png](https://sites.google.com/site/amtmttl/st2/IC7400.png?attredirects=0)  **IC 7400**  ---------------------------------------------------------------------------------------------------  **NOR Gate**  NOR gate is actually a series of OR gate with NOT gate. If we connect the output of an OR gate to the input of  a NOT gate, this combination will work as NOT-OR or NOR gate. Its output is 0 when any or all inputs are 1, otherwise output is 1.  [https://sites.google.com/site/amtmttl/_/rsrc/1470268675875/st2/IC7402.PNG](https://sites.google.com/site/amtmttl/st2/IC7402.PNG?attredirects=0)  **IC 7402**  ---------------------------------------------------------------------------------------------------  **Exclusive OR (X-OR) Gate**  X-OR gate produces an output as 1, when number of 1’s at its inputs is **odd**, otherwise output is 0. It has two inputs and one output.    [https://sites.google.com/site/amtmttl/_/rsrc/1470268675848/st2/IC7486.PNG](https://sites.google.com/site/amtmttl/st2/IC7486.PNG?attredirects=0)  **IC 7486**  ---------------------------------------------------------------------------------------------------  **Exclusive NOR (X-NOR) Gate**  X-NOR gate produces an output as 1, when number of 1’s at its inputs is **not odd**, otherwise output is 0. It has two inputs and one output.  [https://sites.google.com/site/amtmttl/_/rsrc/1470268673514/st2/IC74266.PNG?height=263&width=400](https://sites.google.com/site/amtmttl/st2/IC74266.PNG?attredirects=0)  ---------------------------------------------------------------------------------------------------  **Procedure:**   1. Connect the trainer kit to ac power supply. 2. Connect the inputs of any one logic gate to the logic sources and its output to the logic indicator. 3. Apply varous input combinations and observe output for each one. 4. Verify the truth table for each input/ output combination. 5. Repeat the process for all other logic gates. 6. Switch off the ac power supply.       **Experiment No: 2**  [**GO TO TOP**](https://sites.google.com/site/amtmttl/st2#exp1)  Realization of logic functions with the help of universal gates-NAND Gate.  **Apparatus:** logic trainer kit, NAND gates (IC 7400), wires.  **Theory:**  NAND gate is actually a combination of two logic gates: AND gate followed by NOT gate. So its output is complement of the output of an AND gate.    This gate can have minimum two inputs, output is always one. By using only NAND gates, we can realize all logic functions: AND, OR, NOT, X-OR, X-NOR, NOR. So this gate is also called universal gate.    **NAND gates as NOT gate**  A NOT produces complement of the input. It can have only one input, tie the inputs of a NAND gate together. Now it will work as a NOT gate. Its output is  Y = (A.A)’  =>                                            Y = (A)’  [https://sites.google.com/site/amtmttl/_/rsrc/1470268673214/st2/NAND%20AS%20NOT.PNG](https://sites.google.com/site/amtmttl/st2/NAND%20AS%20NOT.PNG?attredirects=0)  ---------------------------------------------------------------------------------------------------  **NAND gates as AND gate**  A NAND produces complement of AND gate. So, if the output of a NAND gate is inverted, overall output will be that of an AND gate.                                                  Y = ((A.B)’)’  =>                                            Y = (A.B)  [https://sites.google.com/site/amtmttl/_/rsrc/1470268675156/st2/NAND%20AS%20AND.PNG](https://sites.google.com/site/amtmttl/st2/NAND%20AS%20AND.PNG?attredirects=0)  ---------------------------------------------------------------------------------------------------  **NAND gates as OR gate**  From DeMorgan’s theorems: (A.B)’ = A’ + B’  =>                                            (A’.B’)’ = A’’ + B’’ = A + B  So, give the inverted inputs to a NAND gate, obtain OR operation at output.  [https://sites.google.com/site/amtmttl/_/rsrc/1470268674565/st2/NAND%20AS%20OR.PNG](https://sites.google.com/site/amtmttl/st2/NAND%20AS%20OR.PNG?attredirects=0)  ---------------------------------------------------------------------------------------------------  **NAND gates as X-OR gate**  The output of a to input X-OR gate is shown by: Y = A’B + AB’. This can be achieved with the logic diagram shown in the left side.  [https://sites.google.com/site/amtmttl/_/rsrc/1470268675314/st2/NAND%20AS%20XOR.PNG](https://sites.google.com/site/amtmttl/st2/NAND%20AS%20XOR.PNG?attredirects=0)  Gate No.          Inputs                                       Output  1                      A, B                                         (AB)’  2                      A, (AB)’                                  (A (AB)’)’  3                      (AB)’, B                                   (B (AB)’)’  4                      (A (AB)’)’, (B (AB)’)’ A’B + AB’  Now the ouput from gate no. 4 is the overall output of the configuration.  Y         =          ((A (AB)’)’ (B (AB)’)’)’              =          (A(AB)’)’’ + (B(AB)’)’’              =          (A(AB)’) + (B(AB)’)              =          (A(A’ + B)’) + (B(A’ + B’))  =          (AA’ + AB’) + (BA’ + BB’)  =          ( 0 + AB’ + BA’ + 0 )  =          AB’ + BA’  =>                                            Y         =          AB’ + A’B  ---------------------------------------------------------------------------------------------------  **NAND gates as X-NOR gate**  X-NOR gate is actually X-OR gate followed by NOT gate. So give the output of X-OR gate to a NOT gate, overall ouput is  that of an X-NOR gate.                                                  Y = AB+ A’B’  [https://sites.google.com/site/amtmttl/_/rsrc/1470268674421/st2/NAND%20AS%20XNOR.PNG](https://sites.google.com/site/amtmttl/st2/NAND%20AS%20XNOR.PNG?attredirects=0)  ---------------------------------------------------------------------------------------------------  **NAND gates as NOR gate**  A NOR gate is an OR gate followed by NOT gate. So connect the output of OR gate to a NOT gate, overall output is that of a NOR gate.                                                  Y = (A + B)’  [https://sites.google.com/site/amtmttl/_/rsrc/1470268675174/st2/NAND%20AS%20NOR.PNG](https://sites.google.com/site/amtmttl/st2/NAND%20AS%20NOR.PNG?attredirects=0)  ---------------------------------------------------------------------------------------------------  **Procedure:**   1. Connect the trainer kit to ac power supply. 2. Connect the NAND gates for any of the logic functions to be realised. 3. Connect the inputs of first stage to logic sources and output of the last gate to logic indicator. 4. Apply varous input combinations and observe output for each one. 5. Verify the tructh table for each input/ output combination. 6. Repeat the process for all logic functions. 7. Switch off the ac power supply.       **Experiment No: 3**    Realization of logic functions with the help of universal gates-NOR Gate.  **Apparatus:** logic trainer kit, NOR gates (IC 7402), wires.  **Theory:**  NOR gate is actually a combination of two logic gates: OR gate followed by NOT gate. So its output is complement of the output of an OR gate.    This gate can have minimum two inputs, output is always one. By using only NOR gates, we can realize all logic functions: AND, OR, NOT, X-OR, X-NOR, NAND. So this gate is also called universal gate.  ---------------------------------------------------------------------------------------------------  **NOR gates as NOT gate**  A NOT produces complement of the input. It can have only one input, tie the inputs of a NOR gate together. Now it will work as a NOT gate. Its output is  Y = (A+A)’  =>                                            Y = (A)’  **[https://sites.google.com/site/amtmttl/_/rsrc/1470268672936/st2/nor%20as%20not.PNG](https://sites.google.com/site/amtmttl/st2/nor%20as%20not.PNG?attredirects=0)**  ---------------------------------------------------------------------------------------------------  **NOR gates as OR gate**  A NOR produces complement of OR gate. So, if the output of a NOR gate is inverted, overall output will be that of an OR gate.                                                  Y = ((A+B)’)’  =>                                            Y = (A+B)  [https://sites.google.com/site/amtmttl/_/rsrc/1470268675331/st2/nor%20as%20or.PNG](https://sites.google.com/site/amtmttl/st2/nor%20as%20or.PNG?attredirects=0)  ---------------------------------------------------------------------------------------------------  **NOR gates as AND gate**  From DeMorgan’s theorems: (A+B)’ = A’B’  =>                                            (A’+B’)’ = A’’B’’ = AB  So, give the inverted inputs to a NOR gate, obtain AND operation at output.  [https://sites.google.com/site/amtmttl/_/rsrc/1470268673762/st2/nor%20as%20and.PNG](https://sites.google.com/site/amtmttl/st2/nor%20as%20and.PNG?attredirects=0)  ---------------------------------------------------------------------------------------------------  **NOR gates as X-NOR gate**  The output of a two input X-NOR gate is shown by: Y = AB + A’B’. This can be achieved with the logic diagram shown in the left side.  [https://sites.google.com/site/amtmttl/_/rsrc/1470268674344/st2/NOR%20AS%20XNOR.PNG](https://sites.google.com/site/amtmttl/st2/NOR%20AS%20XNOR.PNG?attredirects=0)    **Gate No.          Inputs                                                   Output**  1                      A, B                                                     (A + B)’  2                      A, (A + B)’                                          (A + (A+B)’)’  3                      (A + B)’, B                                          (B + (A+B)’)’  4                      (A + (A + B)’)’, (B + (A+B)’)’            AB + A’B’  Now the ouput from gate no. 4is the overall output of the configuration.  Y         =          ((A + (A+B)’)’ (B +( A+B)’)’)’              =          (A+(A+B)’)’’.(B+(A+B)’)’’              =          (A+(A+B)’).(B+(A+B)’)              =          (A+A’B’).(B+A’B’)  =          (A + A’).(A + B’).(B+A’)(B+B’)  =          1.(A+B’).(B+A’).1  =          (A+B’).(B+A’)                                                              =          A.(B + A’) +B’.(B+A’)                                                              =          AB + AA’ +B’B+B’A’                                                              =          AB + 0 + 0 + B’A’                                                              =          AB + B’A’  =>                                            Y         =          AB + A’B’  ---------------------------------------------------------------------------------------------------  **NOR gates as X-OR gate**  X-OR gate is actually X-NOR gate followed by NOT gate. So give the output of X-NOR gate to a NOT gate, overall ouput is that of an X-OR gate.                                                  Y = A’B+ AB’  [https://sites.google.com/site/amtmttl/_/rsrc/1470268672689/st2/NOR%20AS%20XOR.PNG](https://sites.google.com/site/amtmttl/st2/NOR%20AS%20XOR.PNG?attredirects=0)    ---------------------------------------------------------------------------------------------------  **NOR gates as NAND gate**  A NAND gate is an AND gate followed by NOT gate. So connect the output of AND gate to a NOT gate, overall output is that of a NAND gate.                                                  Y = (AB)’  [https://sites.google.com/site/amtmttl/_/rsrc/1470268674530/st2/nor%20as%20nand.PNG](https://sites.google.com/site/amtmttl/st2/nor%20as%20nand.PNG?attredirects=0)  ---------------------------------------------------------------------------------------------------  **Procedure:**   1. Connect the trainer kit to ac power supply. 2. Connect the NOR gates for any of the logic functions to be realised. 3. Connect the inputs of first stage to logic sources and output of the last gate to logic indicator. 4. Apply varous input combinations and observe output for each one. 5. Verify the tructh table for each input/ output combination. 6. Repeat the process for all logic functions. 7. Switch off the ac power supply.       **Expleriment No:4**  Construction of half adder using XOR and NAND gates and verification of its operation.    **Apparatus:** Logic trainer kit, Logic gates: AND (IC 7408), XOR (IC 7486), NAND(7400).    **Theory:**  A half adder can add two bits at a time. Its outputs are SUM and CARRY. For two bit addition- SUM will be 1, if only one input is 1(X-OR operation). CARRY will be one, when both inputs are 1 (AND operation). So, by using one AND gate and one X-OR gate, a half adder circuit can be constructed. Boolean expressions for the outputs are:                                      SUM = AB’ + A’B                                      CARRY = AB  [https://sites.google.com/site/amtmttl/_/rsrc/1470268674263/st2/HALFADD.PNG](https://sites.google.com/site/amtmttl/st2/HALFADD.PNG?attredirects=0)  **Procedure:**   1. Connect the trainer kit to ac power supply. 2. Connect logic sources to the inputs of the adder. 3. connect output from SUM and CARRY to logic indicators. 4. Apply various input combinations to the adder. 5. Observe the SUM and CARRY outputs, verify the tructh table for each input/ output combination. 6. Switch off the ac power supply.     **Experiment No: 5**  Construction of a NOR gate latch and verification of its operation.    **Apparatus:** Logic trainer kit, NOR gates (IC 7402), wires.  **Theory:**  An S-R (Set, Reset) latch is a digital storage device.  It can store one bit at time. Its output depends upon the combination of inputs and previously stored bit.    An S-R latch can be constructed by using two cross couples NAND/ NOR gates. We will use NOR gates and construct an active high S-R latch.  ---------------------------------------------------------------------------------------------------  [https://sites.google.com/site/amtmttl/_/rsrc/1470268672283/st2/srnor.PNG?height=165&width=400](https://sites.google.com/site/amtmttl/st2/srnor.PNG?attredirects=0)  ---------------------------------------------------------------------------------------------------  **Operation of S-R latch**     1. S=0, R=0: this is the rest state of the NOR latch. This input has no effect on the output state. Outputs (Q, Q’) will remain in whatever state they were prior to the occurance of this input combination. 2. S=1, R=0: this will always set the latch (Q=1, Q’=0), it will remain in this state even after S returns to 0. 3. S=0, R=1: this will always reset the latch (Q=0, Q’=1), it will remain in this state even after r returns to 0. 4. S=1, R=1: this condition tries to set and reset the latch at the same time and produces Q=0, Q’=0.  If inputs are returned to 0 at same time, the resulting output state is unpredictable. This input condition should not be used.   **Procedure:**   1. Connect the trainer kit to ac power supply. 2. Construct an R-S latch by connecting two NOR gates as per logic diagram. 3. Connect logic sources to R, S inputs and outputs Q, Q’ to logic indicators. 4. Apply various R-S combinations and observe Q,Q’  outputs. 5. Verify the truth table. 6. Switch off the ac power supply. |

### Experiments No: 6-11

|  |
| --- |
| **Experiment No:6**  4- bit adder, 2's compliment subtractor circuit using a 4-bit adder IC. Verification of the operation of the circuit.  **Apparatus:** Logic trainer kit, 4-bit adder (IC 7483), X-OR gates (IC 7486), wires  **Theory:** IC 7483 is a 4 bit adder. In binary, subtraction can be performed by using 2's complement method. In this method negative number is converted into its 2's complement and it is added to the other number. The result of this addition is the subtraction of origin numbers.  If we modify the adder circuit, such that 2's complement and simple representation are presented, we can perform addition subtraction as required. X-OR gate is used as a controlled inverter/ buffer for this purpose. Use it as buffer for addition and inverter for subtraction.  [https://sites.google.com/site/amtmttl/_/rsrc/1470268674611/st3/addsub.GIF?height=315&width=400](https://sites.google.com/site/amtmttl/st3/addsub.GIF?attredirects=0)  **Procedure:**  1. Connect the IC 7483 and IC 7486 as per diagram.  2. Connect all A's and all B's to logic sources, S's to logic indicators.  3. Connect Cin to logic 0, this will set the circuit for addition.  4. Give various input combinations, verify adder operation. Here Cout is MSB of addition.  5. Connect Cin to logic 1, this will set the circuit for subtraction by 2's complement method.  6. Give various input combinations and observe outputs.  Here Cout is neglected (2's complement subtraction)  7. Switch off power supply.  **Experiment No:7**  **Aim:** Verification of truth table for 7 segment decoder/ driver ICs  **Apparatus:** Logic trainer kit, 7 segment decoder/ driver (IC 7447), wires  **Theory:** Seven Segment is a display device. 7 LEDs are used in this device. When a LED is forward biased, it emits light. By using a 7 segment, we can display various characters, formed by forward biased segments.  It can be used to display 0-9 and a-F, BCD and Hexadecimal numbers can be displayed with it.  7 segment decoder/ driver is a combination of decoder circuit and display driver (for 7 segments). Input is given from 4 inputs and output is shown on display  **[https://sites.google.com/site/amtmttl/_/rsrc/1470268676277/st3/7seg.JPG](https://sites.google.com/site/amtmttl/st3/7seg.JPG?attredirects=0)**  **Procedure:**  1. Connect 4 inputs of display/ driver circuit to logic sources and switch on main supply.  2. Give any combination to circuit.  3. Observe the display; it should be according to BCD/ Hexadecimal encoding.  4. Give various input combinations, observe their corresponding outputs.  5. Connect Cin to logic 1, this will set the circuit for subtraction by 2's complement method.  6. Switch off power supply.  **Experiment No: 8**  **Aim**  verfification of truth table for 8:1 multiplexer.  **Apparatus:** Logic trainer kit, 8:1 multiplexer (IC 74151), wires  **Theory:** A multiplexer (MUX) is an electronic circuit which has many inputs  but only one output. It has some select lines, number of select lines is related to the number of inputs. If there are N select lines, any one out of  2N inputs can be selected. It is actually a decoder with all AND gates connected to separate select combination and a unique input line. Their outputs are given to an OR gate to obtain one output. By using proper combination of select lines, any one input can be selected at a time and its data is sent to the output.  [https://sites.google.com/site/amtmttl/_/rsrc/1470268675678/st3/MUX.GIF](https://sites.google.com/site/amtmttl/st3/MUX.GIF?attredirects=0)  **Procedure:**  1. Connect inputs and select lines of multiplexer to logic sources.  2. Connect output to logic indicator.  3. Set/ reset the inputs  value in desired manner.  4. By using select lines give any combination to multixer  5. Observe the output and verify that it is same as input given to the selected input.  6. Give various input combinations, observe their corresponding outputs.  7. Switch off power supply.  **Experiment No:9**  Construction of 4 bit SISO, SIPO, PISO, PIPO shift registers and verification of their operation.  **Apparatus:** Logic trainer kit, D flip flop (IC 74143), wires  **Theory:** Shift register is used to move the data. To move data, it must be stored. So shift register actually stores data and moves it to left, right as per signal given to it. Its various types are:  -Serial In Serial Out  -Serial In Parallel Out  -Parallel In Serial Out  -Parallel In Parallel Out  As flip flops are capable to store data (1 bit in a flip flop), they are used to construct shift registers  **Serial In:** Output of one flip flop is input of another. Data is serially given i.e. only first flip flop receives data; it is shifted to next flip flops.  **Serial Out:** Data is taken out from last flip flop  **Parallel In:** All flip flops are loaded simultaneously  **Parallel Out:** data is taken parallely by taking outputs from all flip flops at same time.  **[https://sites.google.com/site/amtmttl/_/rsrc/1470268673384/st3/sisopo.GIF](https://sites.google.com/site/amtmttl/st3/sisopo.GIF?attredirects=0)**  **[https://sites.google.com/site/amtmttl/_/rsrc/1470268673481/st3/PIPO.GIF](https://sites.google.com/site/amtmttl/st3/PIPO.GIF?attredirects=0)**  **[https://sites.google.com/site/amtmttl/_/rsrc/1470268675950/st3/PIsO.GIF](https://sites.google.com/site/amtmttl/st3/PIsO.GIF?attredirects=0)**  **Procedure:**  1. Connect flip flops as per given diagram  2. Connect inputs to Q3, and Q0 to logic indicator.  3. Apply clock and data train to Q3, observe output at Q0(SISO)  4. For SIPO, observe outputs at all Q’s by connecting all to logic indicators  5. Repeat for parallel in by connecting D’s to logic sources and outputs at Q0 for PISO, Q’s for PISO  6. Switch off supply.  **Experiment No 10**  Construction and verification of operation of 4-bit ring counter  **Apparatus:** Logic trainer kit, D flip flops(IC 74173), wires.  **Theory**: Ring counter is constructed by modifying the Serial In Serial Out shift register. The basic ring counter can be obtained by connecting the last output to first input. When clock signal is applied, data is shifted in a circular manner or in a closed ring, so it is called a ring counter.  [https://sites.google.com/site/amtmttl/_/rsrc/1470268676142/st3/ringcount.GIF](https://sites.google.com/site/amtmttl/st3/ringcount.GIF?attredirects=0)  **Procedure:**  1. Connect d flip flops as per circuit diagram.  2. Connect clock signal to logic source  3. Apply clock signal continuously and observe the output  4. If all outputs are zero then disconnect input to first flip flop  5. Apply 1 to it and give clock signal.  6. Reconnect Q0 to D3 and check operation of this circuit  7. Observe the out put.  8. Switch off power supply  **Experiment No 11**  Construction and verification of operation of 4-bit ring counter  **Apparatus:** Logic trainer kit, JK flip flop, wires  **Theory**: A digital counter is an electronic circuit which is used to count the clock pulses. In a counter, a flip flop in toggle mode is used. So JK flip flops can be used with JKs connected to 1 or T flip flops can be used.  Counter can be up or down counter. When output is in the form of increasing binary number, it is an up counter, otherwise down counter. Mod of a counter is the number of outputs i.e. count values. A Mod –M counter can count from 0 to M-1 and it requires log2m flip flops.  [https://sites.google.com/site/amtmttl/_/rsrc/1470268674067/st3/counter4.GIF](https://sites.google.com/site/amtmttl/st3/counter4.GIF?attredirects=0)  [https://sites.google.com/site/amtmttl/_/rsrc/1470268673248/st3/counterW.GIF](https://sites.google.com/site/amtmttl/st3/counterW.GIF?attredirects=0)  **Procedure:**  1. Connect the flip flops as per diagram  2. Connect clock to logic source, outputs to logic indicators, j and k to 1.  3. Start giving clock pulses (on/off repetition), it will change the output.  4. Observe the outputs and compare with binary count.  5. Verify its operation.  6. Switch off power supply |