Ramakrishna Mission
Vivekananda Centenary College, Rahara

Name : Darpan Bhattacharya

Course : B.Sc. Computer Science (Hons.)

Semester : 2nd Semester

Roll No. : 715

Reg. No. : A01-1112-117-014-2021 of 2021-2022

Subject : Electronics Practical (GE2)

Session : 2021 – 2022

DIGITAL ELECTRONIC CIRCUITS

Experiment 1

Aim

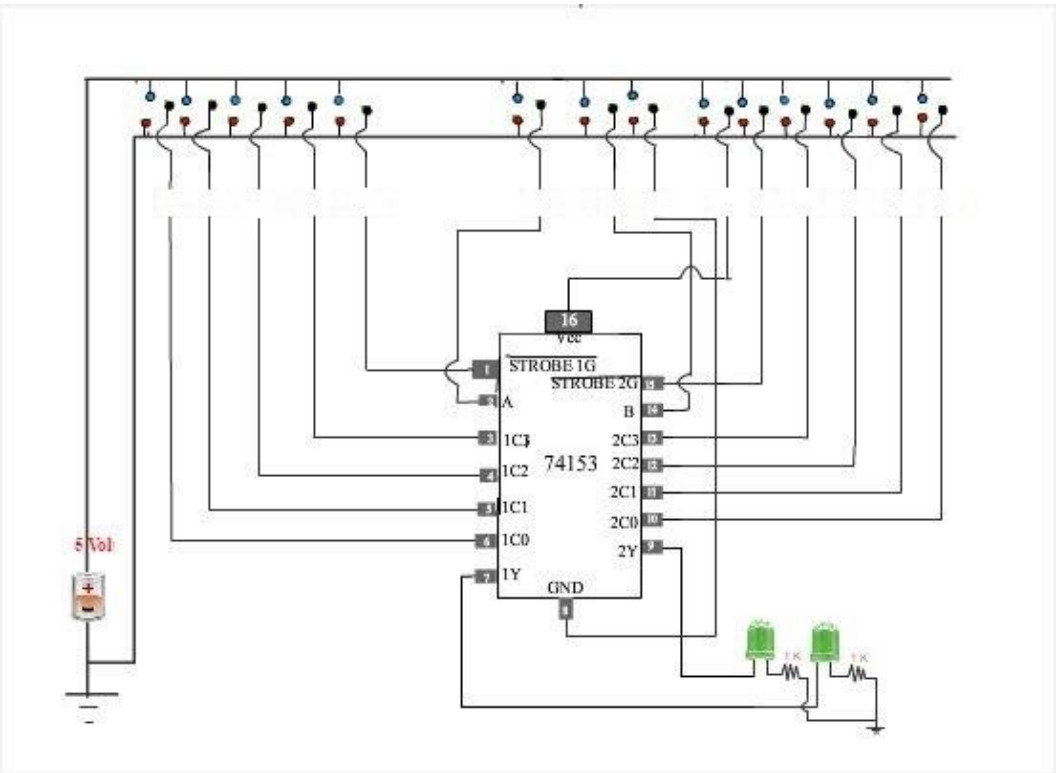
To verify and understand the functionality of a 4x1 multiplexer.

Theory

A multiplexer is a combinational circuit that receives binary information from one of 2^n input data lines and directs it to a single output line. The selection of a particular input data line for the output is determined by a set of selection inputs. A 2^n-to-1 multiplexer has 2^n input data lines and n input selection lines whose bit combinations determine which input data are selected for the output.

74153 is a 4x1 dual multiplexer. It has 2 select lines, A and B. The input lines are $1I_0$, $1I_1$, $1I_2$, $1I_3$, and $2I_0$, $2I_1$, $2I_2$, $2I_3$. The output lines are Y_0 and Y_1.

Circuit diagram



Experimental Results

The experiment was performed in http://vlabs.iitkgp.ac.in/dec/exp4/exp4a/exp4p1.html, and the following truth table was obtained:

TRUTH TABLE

Multiplxer 2 is active					Multiplxer 1 is active				
Strobe		Select Lines		Output	Strobe		Select Lines		Output
1G	2G	A	B	1Y	1G	2G	A	B	2Y
1	1	x	x	0	1	1	x	x	0
0	1	0	0	1C0	1	0	0	0	2C0
0	1	0	1	1C1	1	0	0	1	2C1
0	1	1	0	1C2	1	0	1	0	2C2
0	1	1	1	1C3	1	0	1	1	2C3

Discussion

This experiment was performed using the Digital Electronics Circuits Laboratory of Virtual Labs. The experimental data was verified and the results were similar to the 4x1 dual multiplexer studied theoretically.

Similarly, we can implement 8x1 multiplexer using 74151 IC. Using 74151 IC, we can also implement other combinational circuits such as half adder, full adder, half subtractor and full subtractor.

Experiment 2

Aim

To verify and understand the functionality of a S-R flip-flop.

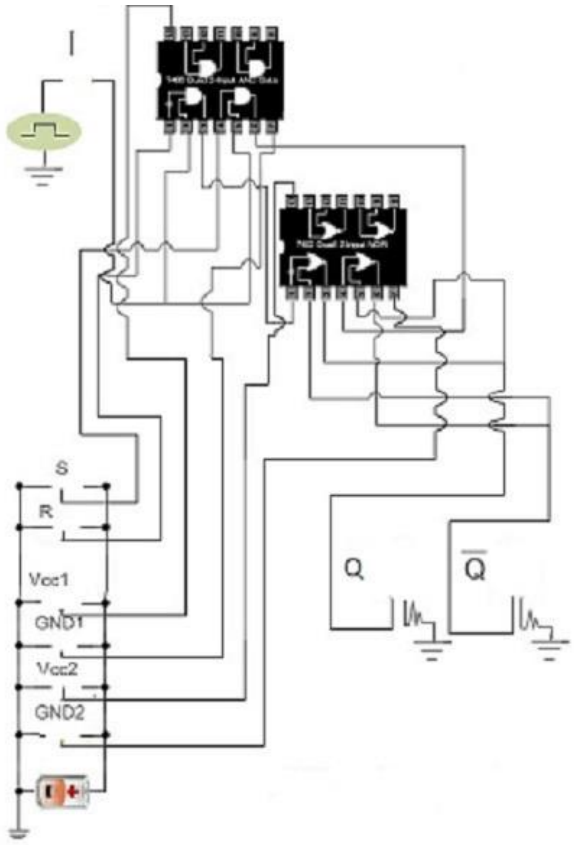
Theory

The logic circuits whose outputs at any instant of time depend not only on the present input but also on the past outputs are called sequential circuits.

The simplest kind of sequential circuit which is capable of storing one bit of information is called latch. The operation of basic latch can be modified, by providing an additional control input that determines, when the state of the circuit is to be changed.

The latch with additional control input is called the flip-flop. The additional control input is either the clock or enable input.

Circuit diagram



Experimental Results

The experiment was performed in http://vlabs.iitkgp.ac.in/dec/exp8/exp8a/exp8p1.html, and the following truth table was obtained:

Plot | TRUTH TABLE | Add to Table

Clock	S	R	Q	Q_{t+1}	Action
1	0	0	0	0	No change
1	1	0	1	0	Set
1	0	1	0	1	Reset
1	1	1	1	?	Forbidden

Discussion

This experiment was performed using the Digital Electronics Circuits Laboratory of Virtual Labs. The experimental data was verified and the results were similar to the S-R flip-flop studied theoretically.

Experiment 3

Aim

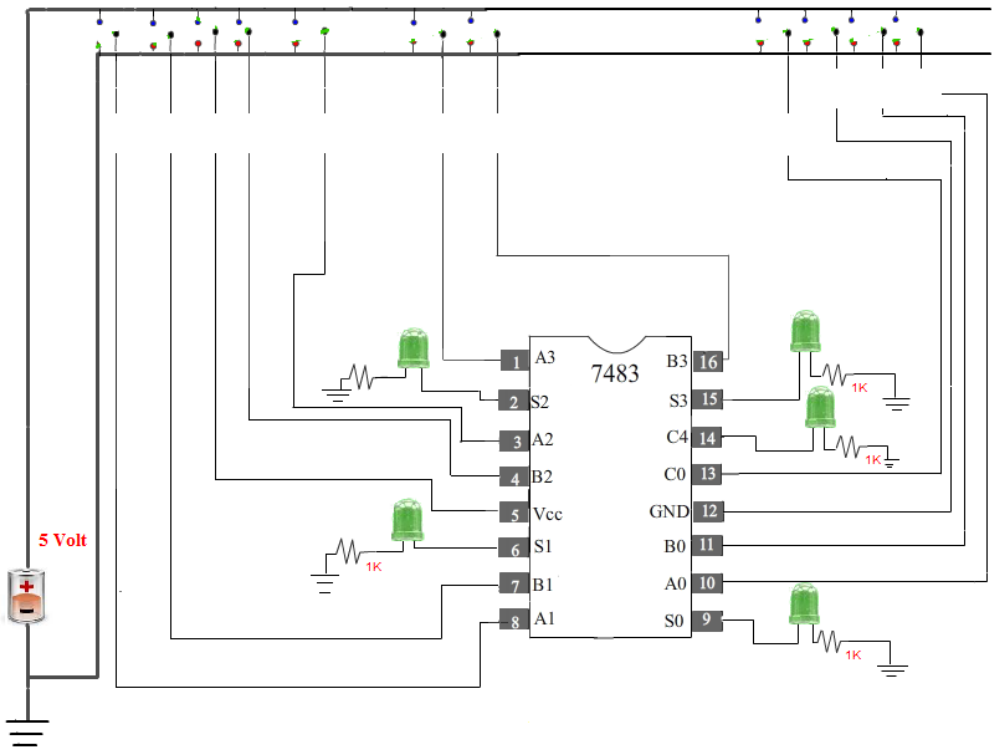
To verify and understand the functionality of a 4-bit binary full adder.

Theory

The most basic digital arithmetic circuit is the addition of two binary digits. A combinational circuit that performs the arithmetic addition of two bits is called a half-adder. One that performs the addition of three bits (two significant bits and a previous carry) is called a full-adder. The name of the former stems from the fact that two half-adders are needed to implement a full-adder.

IC 7483 is a 4-bit binary full adder which accepts two 4-bit binary words $A_3A_2A_1A_0$ and $B_3B_2B_1B_0$ and a carry input (C_0) as inputs and produces a 4-bit binary sum output $S_3S_2S_1S_0$ and a carry output C_4.

Circuit diagram



Experimental Results

The experiment was performed in http://vlabs.iitkgp.ac.in/dec/exp7/exp7p1/exp7p1.html, and the following truth table was obtained:

TRUTH TABLE

A				B				Sum				
A_3	A_2	A_1	A_0	B_3	B_2	B_1	B_0	C_4	S_3	S_2	S_1	S_0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	1	0	0	0	1	0
0	0	0	1	0	1	0	1	0	0	1	1	0
0	0	0	1	0	1	1	1	0	1	0	0	0
0	1	0	1	0	1	1	1	0	1	1	0	0
0	1	1	1	0	1	1	1	0	1	1	1	0
0	1	1	1	0	1	0	1	0	1	1	0	0
0	1	1	1	1	1	0	1	1	0	1	0	0
1	1	1	1	1	1	0	1	1	1	1	0	0
1	1	1	1	1	1	1	1	1	1	1	1	0

Discussion

This experiment was performed using the Digital Electronics Circuits Laboratory of Virtual Labs. The experimental data was verified and the results were similar to the 4-bit binary full-adder studied theoretically.

Experiment 4

Aim

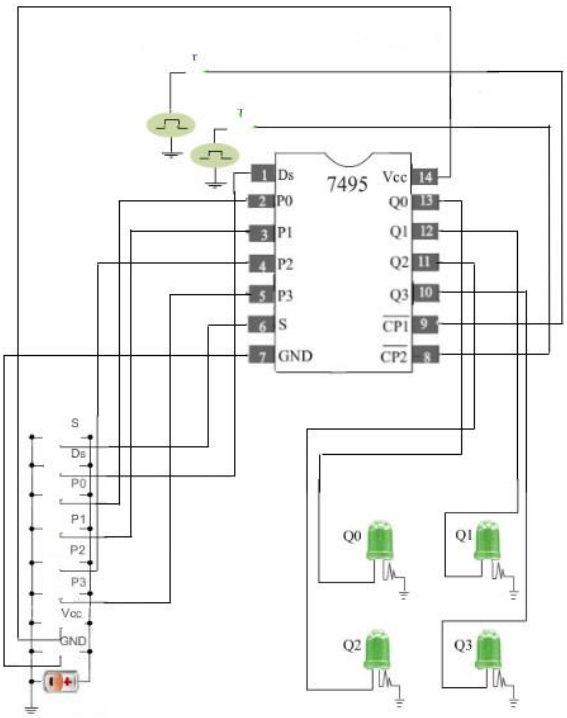
To verify and understand the functionality of a shift register.

Theory

A register is a group of flip-flops with each flip-flop capable of storing one bit of information. An n-bit register has a group of n flip-flops and is capable of storing any binary information of n bits. In addition to the flip-flops, a register may have combinational gates that perform certain data-processing tasks. In its broadest definition, a register consists of a group of flip-flops and gates that effect their transition. The flip-flops hold the binary information and the gates control when and how new information is transferred into the register.

A register capable of shifting information either to right or left is called a shift register. In a shift register, the flip-flops are connected in such a way that the binary bis are entered into the shift register, shifted from one location to another and finally shifted out. The different types of shift registers are: Serial-in-Serial-out (SISO), Serial-in-Parallel-out (SIPO), Parallel-in-Serial-out (PISO), and Parallel-in-Parallel-out (PIPO).

Circuit diagram



Experimental Results

The experiment was performed in http://vlabs.iitkgp.ac.in/dec/exp9/exp9a/exp9a.html, and the following truth table was obtained:

Clock pulse	Serial Input	Flip-flops				Serial output
C_p	D_s	Q_3	Q_2	Q_1	Q_0	Q_0
0	1	0	0	0	0	0
1	1	1	0	0	0	0
1	0	1	1	0	0	0
1	1	0	1	1	0	0
1	-	1	0	1	1	1
1	-	0	1	0	1	1
1	-	0	0	1	0	0
1	-	0	0	0	1	1
1	-	0	0	0	0	0

Discussion

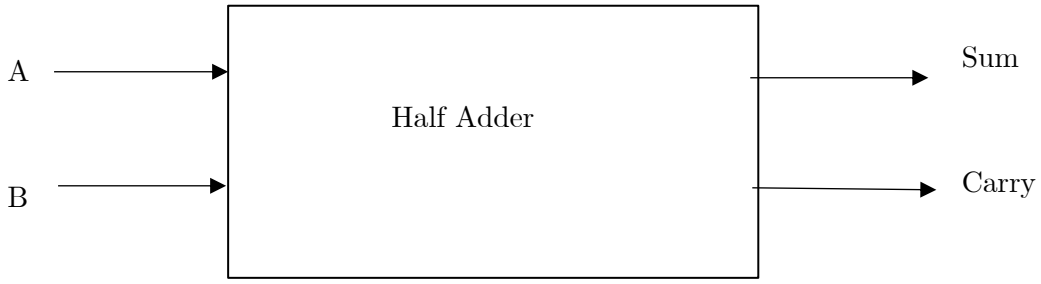
This experiment was performed using the Digital Electronics Circuits Laboratory of Virtual Labs. The experimental data was verified and the results were similar to the shift register studied theoretically.

PROGRAMMING IN VHDL

Half Adder

A half adder is a combinational circuit that performs the arithmetic addition of two binary digits. The input variables of the half adder are called the augend and addend bits. The output variables are called sum and carry.

Block diagram

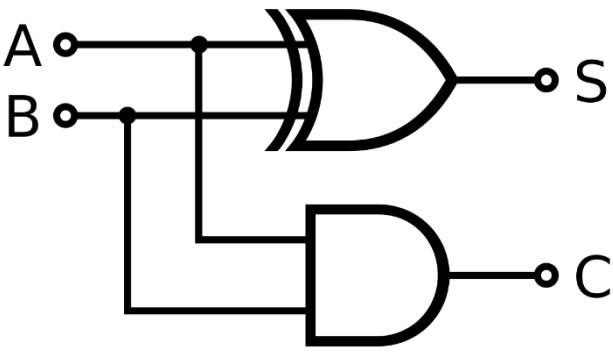


Here, Sum = A'B + AB' and Carry = A.B

Truth table

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Circuit Diagram



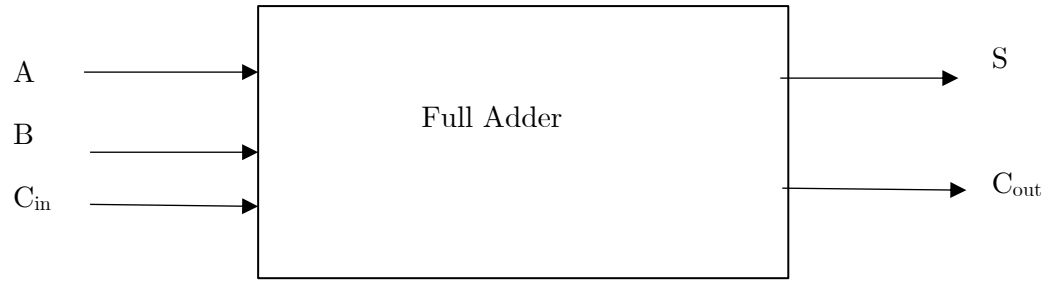
VHDL Program

```
Entity HalfAdder is
port(A,B: in std_logic;
     S,C: out std_logic);
end HalfAdder;

Architecture ArchHA of HalfAdder is
begin
    S <= ((not A) and B) or (A and (not B));
    C <= A and B;
end ArchHA;
```

Full Adder

A full adder is a combinational circuit that forms the arithmetic sum of three input bits. It consists of three inputs and two outputs. Two of the input variables, denoted by A and B, represent the two significant bits to be added. The third input, C_{in}, represents the carry from the previous lower significant position. The sum output is denoted by S, while the carry output is denoted by C_{out}.

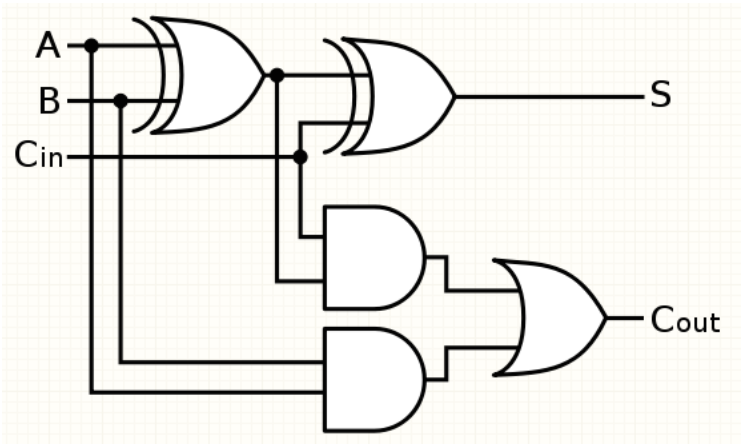
Block diagram



Truth table

A	B	C_{in}	Sum	C_{out}
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

Circuit Diagram



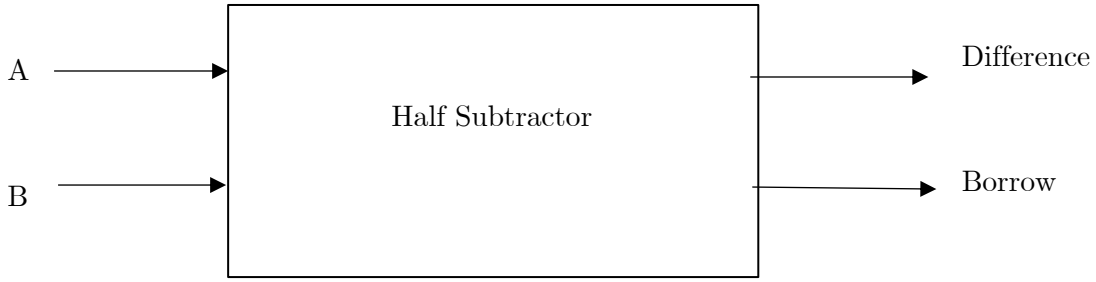
VHDL Program

```
Entity FullAdder is
port(A,B,Cin: in std_logic;
     S,Cout: out std_logic);
end FullAdder;

Architecture ArchFA of FullAdder is
begin
    S <= A xor B xor Cin;
    Cout <= (A and B) or (B and Cin) or (A and Cin);
end ArchFA;
```

Half Subtractor

A half subtractor is a combinational circuit that performs the arithmetic subtraction of two binary digits. The input variables of the half subtractor are denoted by A (minuend) and B (subtrahend) and the output variables are denoted by Difference and Borrow.

Block diagram

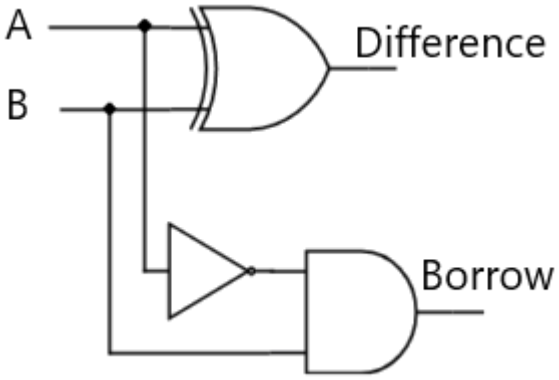


Here, Difference = A $\oplus$ B and Carry = A'B

Truth table

A	B	Difference	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Circuit Diagram



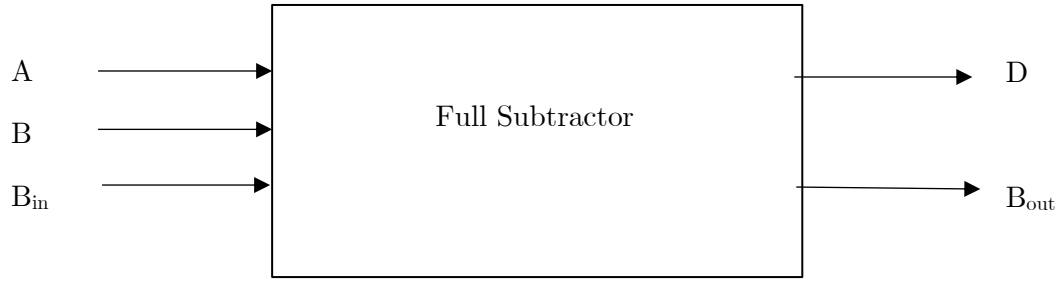
VHDL Program

```
Entity HSub is
Port(A,B : in std_logic;
     D,Br : out std_logic);
end HSub;

Architecture ArchHS of HSub is
begin
    D <= A xor B;
    Br <= A and (not B);
end ArchHS;
```

Full Subtractor

A full subtractor is a combinational circuit that performs the arithmetic subtraction involving three binary digits. The input variables of the full subtractor are denoted by A (minuend), B (subtrahend), B_{in} (borrow - in) and the output variables are denoted by D (difference) and B_{out} (borrow - out).

Block diagram

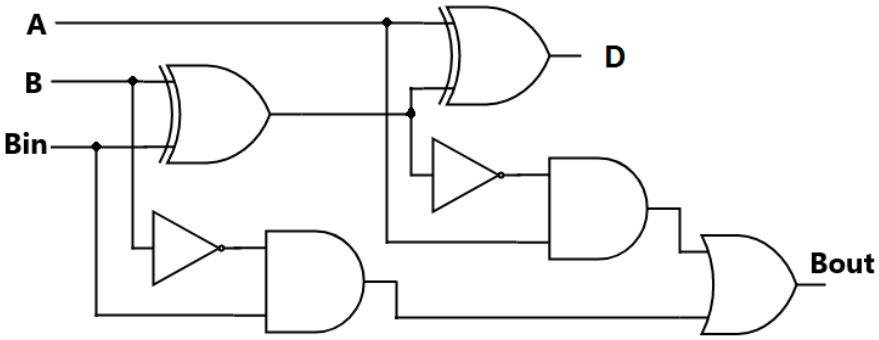


Here, $D = A \oplus B \oplus C$ and $B_{out} = A'B_{in} + A'B + B_{in}B$

Truth table

A	B	B_{in}	Difference	B_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Circuit Diagram



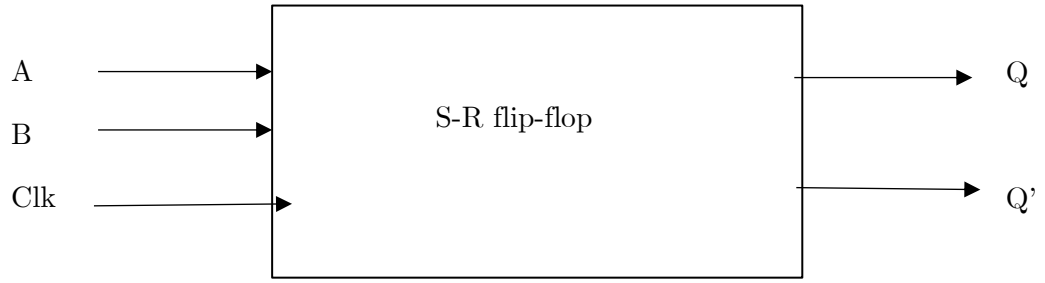
VHDL Program

```
Entity FullSub is
Port(A,B,C : in std_logic;
     D,Br : out std_logic);
end FullSub;

Architecture ArchFS of FullSub is
begin
    D <= A xor B xor C;
    Br <= (A and B) or (B and (not C)) or (A and (not C));
end ArchFS;
```

S-R flip-flop

A S-R flip-flop is one of the most basic sequential logic circuits. A flip-flop is a one – bit memory device that switches states when directed by the clock pulse. It has three inputs, labelled S (for set), R (for reset), and C (for clock). It has an output Q and a complemented output Q'.

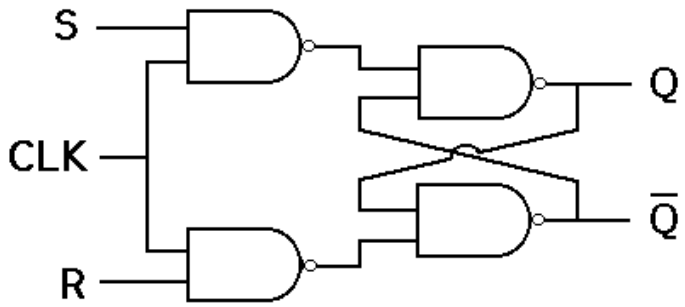
Block diagram



Truth table

A	B	Q_n	Q_{n+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	indeterminate
1	1	1	indeterminate

Circuit Diagram



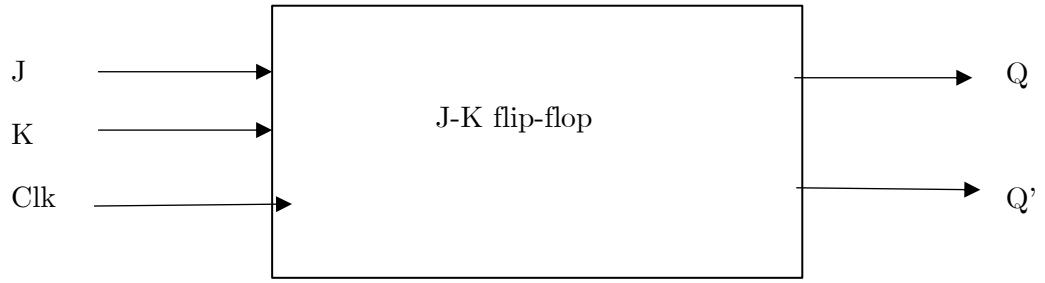
VHDL Program

```
Entity SRFlipFlop is
Port(S,R,Clk : in std_logic;
     Q, Qbar : out std_logic);
end SRFlipFlop;

Architecture ArchSRFF of SRFlipFlop is
signal temp : std_logic;
begin
process(Clk);
begin
    if(Clk'event and Clk = '1') then
        if(S = '0' and R = '0') then
            temp <= temp;
        elsif(S = '0' and R = '1') then
            temp <= '0';
        elsif(S = '1' and R = '0') then
            temp <= '1';
        end if;
    end if;
end process;
Q <= temp;
Qbar <= not temp;
end ArchSRFF;
```

J-K flip-flop

A J-K flip-flop is a refinement of the S-R flip-flop in which the indeterminate condition of the SR type is defined in the JK type. Inputs J and K behave like inputs S and R to set and clear the flip-flop, respectively. When inputs J and K are both equal to 1, a clock transition switches the outputs of the flip-flop to their complement state.

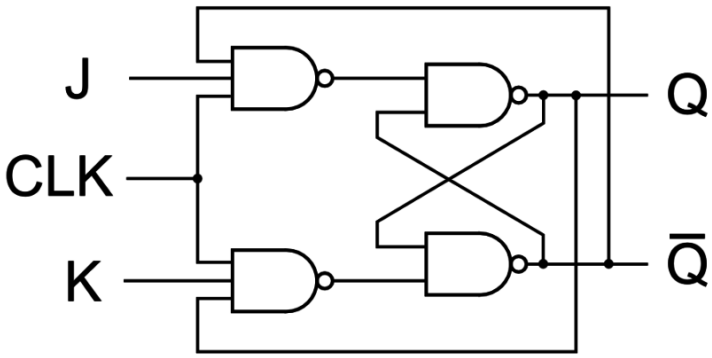
Block diagram



Truth table

A	B	Q_n	Q_{n+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Circuit Diagram



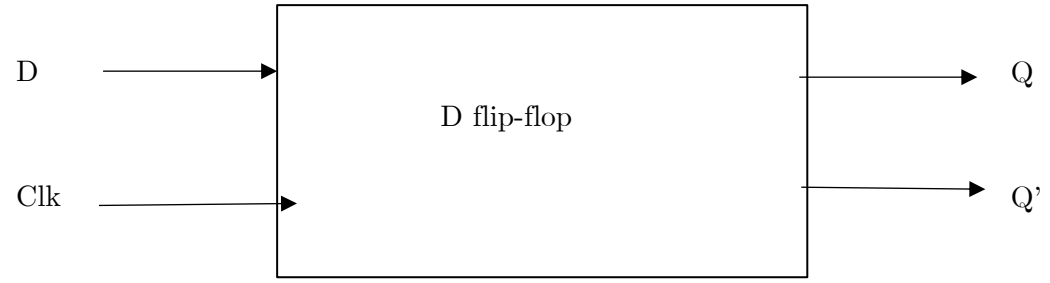
VHDL Program

```
Entity JKFlipFlop is
Port(J,K,Clk : in std_logic;
     Q, Qbar : out std_logic);
end JKFlipFlop;

Architecture ArchJKFF of JKFlipFlop is
signal temp : std_logic;
begin
process(Clk);
begin
    if(Clk'event and Clk = '1') then
        if(J = '0' and K = '0') then
            temp <= temp;
        elsif(J = '0' and K = '1') then
            temp <= '0';
        elsif(J = '1' and K = '0') then
            temp <= '1';
        else
            temp <= not temp;
        end if;
    end if;
end process;
Q <= temp;
Qbar <= not temp;
end ArchJKFF;
```

D flip-flop

A D (data) flip-flop is a slight modification of the S-R flip-flop. An S-R flip-flop is converted to a D flip-flop by inserting an inverter between S and R and assigning the symbol D to the single input. The D input is sampled during the occurrence of a clock transition from 0 to 1. If D = 1, the output of the flip-flop goes to the 1 state, but if D = 0, the output of the flip-flop goes to the 0 state.

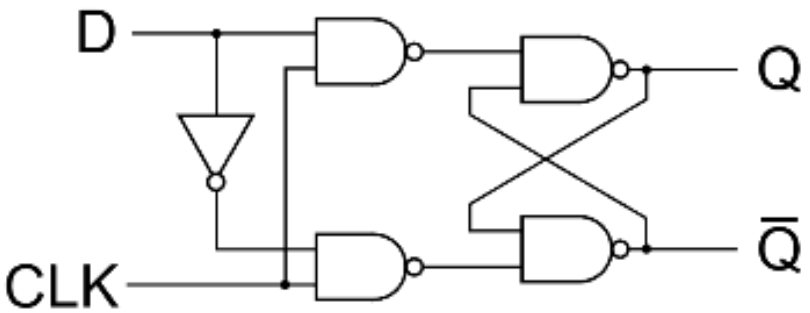
Block diagram



Truth table

D	Q_n	Q_{n+1}
0	0	0
0	1	0
1	0	1
1	1	1

Circuit Diagram



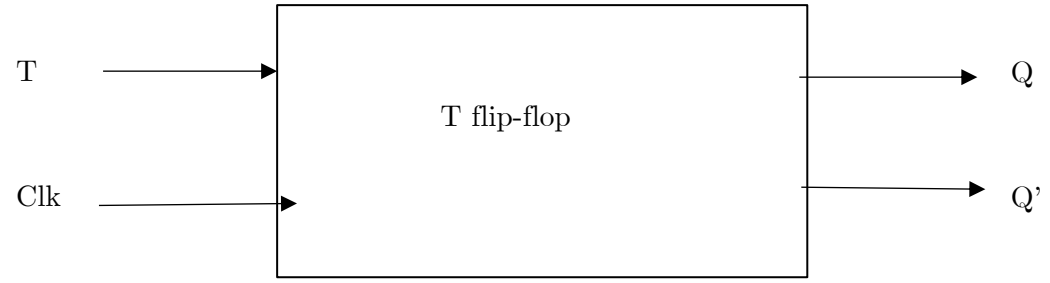
VHDL Program

```
Entity DFlipFlop is
Port(D,Clk : in std_logic;
     Q,Qbar : out std_logic);
end DFlipFlop;

Architecture ArchDFF of DFlipFlop os
signal temp : std_logic;
begin
process(Clk)
begin
    if(Clk'event and Clk = '1') then
        if(D = '0') then
            temp <= '0';
        else
            temp <= '1';
        end if;
    end if;
end process;
Q <= temp;
Qbar <= not temp;
end ArchDFF;
```

T flip-flop

A T (toggle) flip-flop is obtained from a J-K flip flop when inputs J and K are connected to provide a single input designated by T. The T flip-flop therefore has only two conditions. When T = 0 (J = K = 0) a clock transition does not change the state of the flip-flop. When T = 1 (J = K = 1) a clock transition complements the state of the flip-flop.

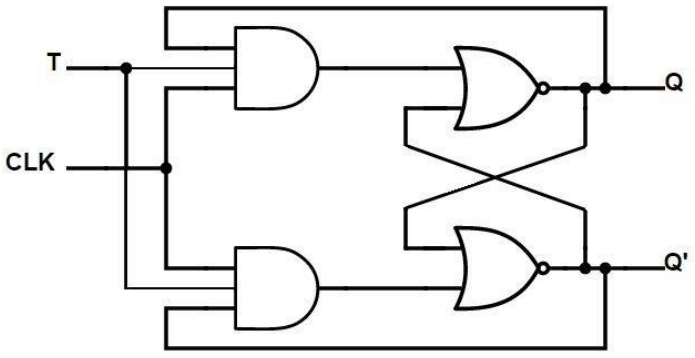
Block diagram



Truth table

Q_n	T	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

Circuit Diagram



VHDL Program

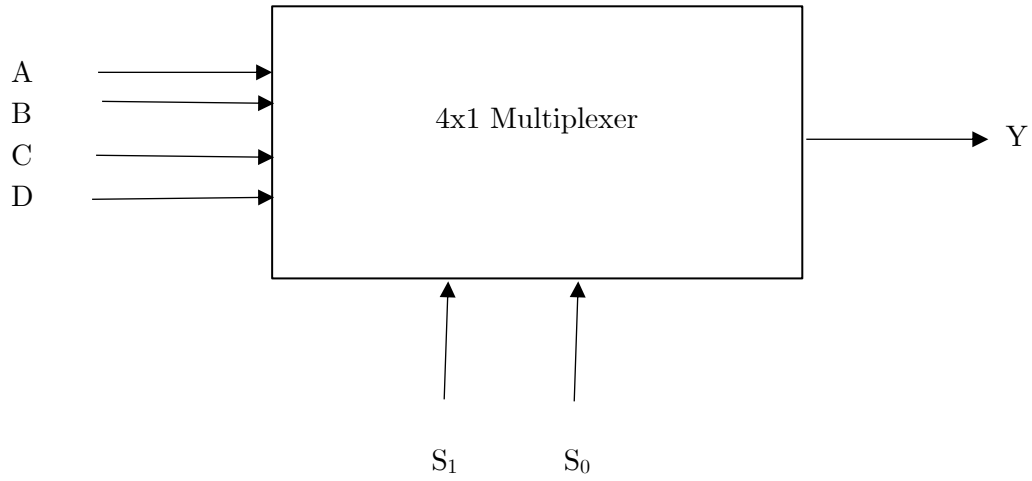
```
Entity TFlipFlop is
Port(T,Clk : in std_logic;
     Q,Qbar : out std_logic);
End TFlipFlop;

Architecture ArchTFF of TFlipFlop is
signal temp : std_logic;
begin
process(Clk)
begin
    if(Clk'event and Clk = '1') then
        if(T = '0') then
            temp <= temp;
        else
            temp <= not temp;
        end if;
    end if;
end process;
Q <= temp;
Qbar <= not temp;
end ArchTFF;
```

4x1 Multiplexer

A multiplexer is a combinational circuit that receives binary information from one of 2^n input data lines and directs it to a single output line. The selection of a particular input data line for the output is determined by a set of selection inputs.

A 4x1 multiplexer has 4 data inputs, 2 selection lines and 1 output line.

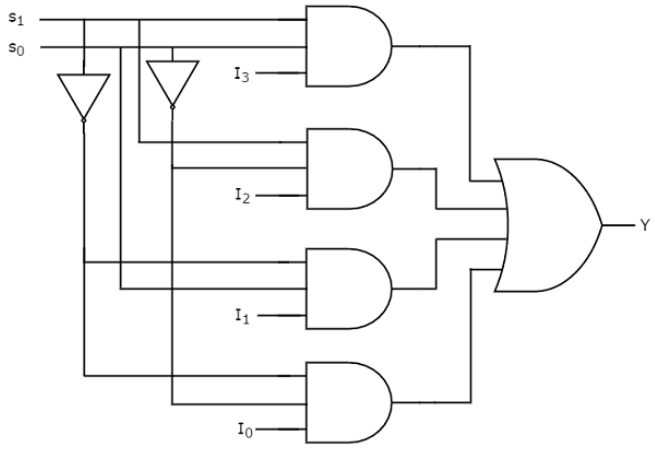
Block diagram



Truth table

S_1	S_0	Y
0	0	Y=A
0	1	Y=B
1	0	Y=C
1	1	Y=D

Circuit Diagram



Here I_0, I_1, I_2, and I_3 can be replaced by A, B, C and D respectively.

VHDL Program

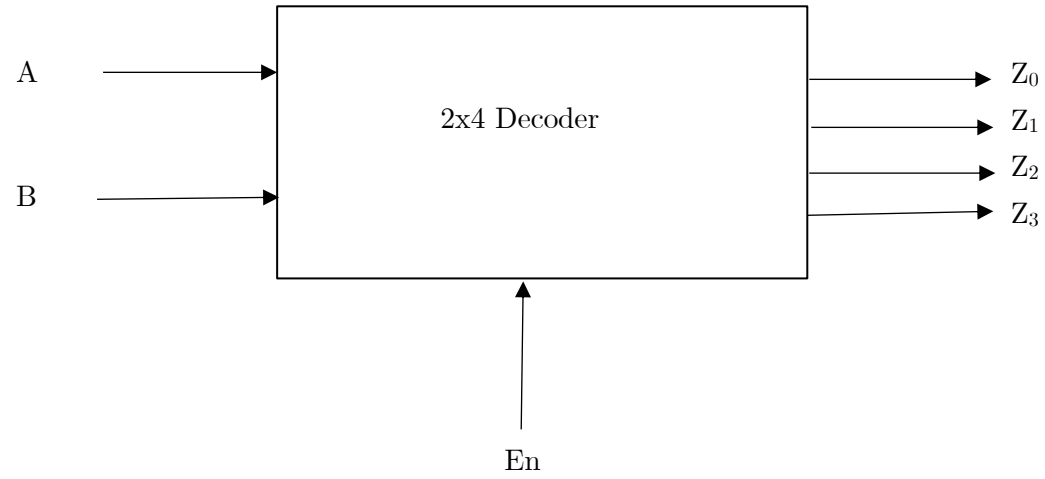
```
Entity Mux4x1 is
Port(S : in std_logic_vector(1 downto 0);
     I : in std_logic_vector(3 downto 0);
     Y : out std_logic);
End Mux4x1;

Architecture ArchMux4x1 of Mux4x1 is
begin
process(S,I)
begin
case S is
    when ‘‘00’’ =>
        Y <= I(0);
    when ‘‘01’’ =>
        Y <= I(1);
    when ‘‘10’’ =>
        Y <= I(2);
    when ‘‘11’’ =>
        Y <= I(3);
end case;
end process;
end ArchMux4x1;
```

2x4 Decoder

A multiplexer is a combinational circuit that receives binary information from one of 2^n input data lines and directs it to a single output line. The selection of a particular input data line for the output is determined by a set of selection inputs.

A 4x1 multiplexer has 4 data inputs, 2 selection lines and 1 output line.

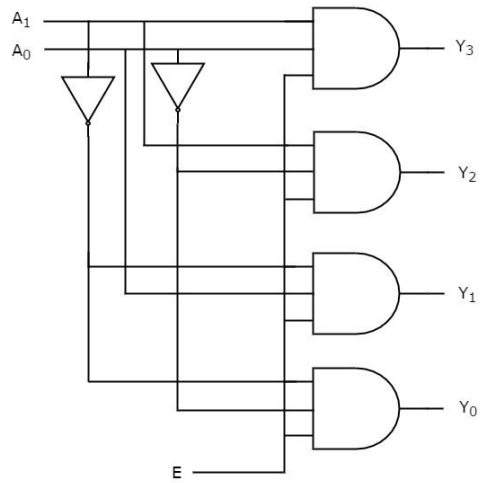
Block diagram



Truth table

A	B	En	Z_3	Z_2	Z_1	Z_0
0	0	1	0	0	0	1
0	1	1	0	0	1	0
1	0	1	0	1	0	0
1	1	1	1	0	0	0
x	x	0	0	0	0	0

Circuit Diagram



Here Y_0, Y_1, Y_2, and Y_3 can be replaced by Z_0, Z_1, Z_2 and Z_3 respectively, and A_0 and A_1 can be replaced by A and B respectively.

VHDL Program

```
Entity Decoder2x4 is
Port(A,B,En : in std_logic;
     Z : out std_logic_vector(3 downto 0));
end Decoder2x4;

Architecture ArchD2x4 of Decoder2x4 is
    signal ABAR, BBAR : std_logic;
    begin
        ABAR <= not A;
        BBAR <= not B;
        Z(0) <= ABAR and BBAR and En;
        Z(1) <= ABAR and B and En;
        Z(2) <= A and BBAR and En;
        Z(3) <= A and B and En;
end ArchD2X4;
```