**EXPERIMENT 8**

Rishabh Shah J058

**1)**

//master receiving

#include<Wire.h>

void setup()

{

Wire.begin();

Serial.begin(9600);

}

void loop()

{

Wire.requestFrom(4, 6); // (addr,bytes)

while(Wire.available())

{

char c = Wire.read();

Serial.print(c);

}

delay(500);

}

//slave

#include<Wire.h>

void setup()

{

Wire.begin(4);

Serial.begin(9600);

Wire.onRequest(transmitData);

}

byte x = 0;

void transmitData()

{

Wire.write(“x is =”);

Wire.write(x);

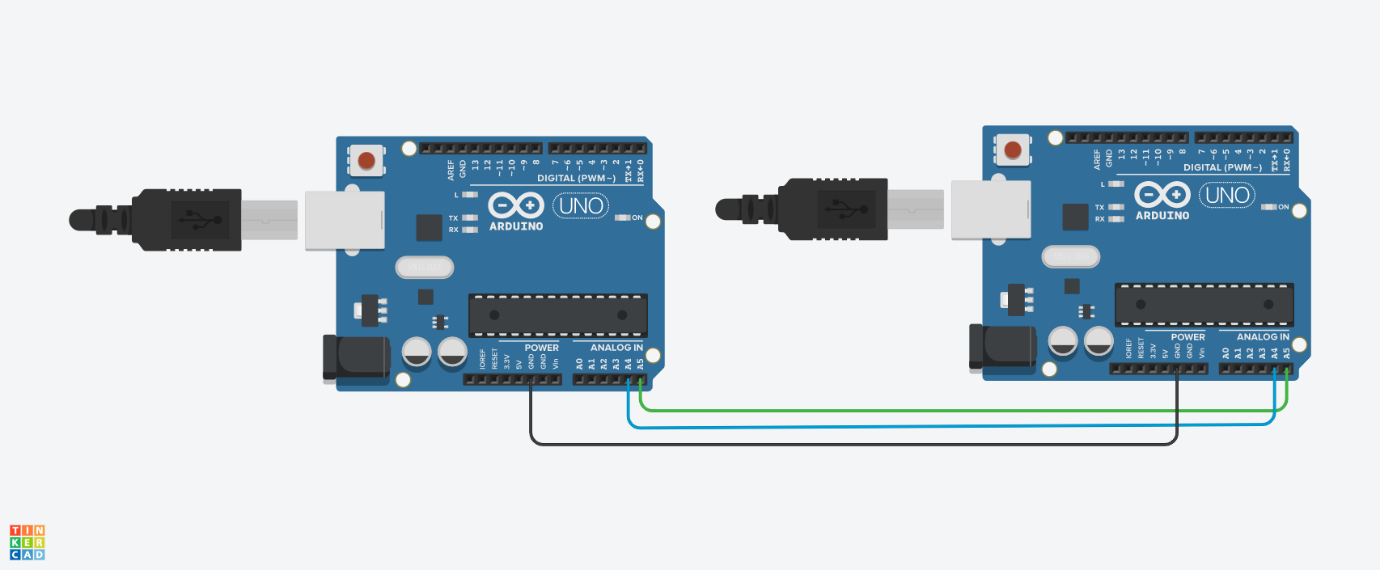
x++;

}

void loop()

{

}



**2)**

//master sender

#include<Wire.h>

int ADDR = 4;

void setup()

{

Wire.begin(); // join i2c bus (address optional for master)

}

byte x = 0;

void loop()

{

Wire.beginTransmission(ADDR);

Wire.write(“x is =”); //5bytes

Wire.write(x);//1byte

Wire.endTransmission();

x++;

delay(100);

}

//slave receiver

#include<Wire.h>

void setup()

{

Wire.begin(4);

Wire.onReceive(receiveData);

Serial.begin(9600);

}

void receiveData()

{

while(1 < Wire.available())

{

Char c = Wire.read();

Serial.print(c);

}

byte a = Wire.read();

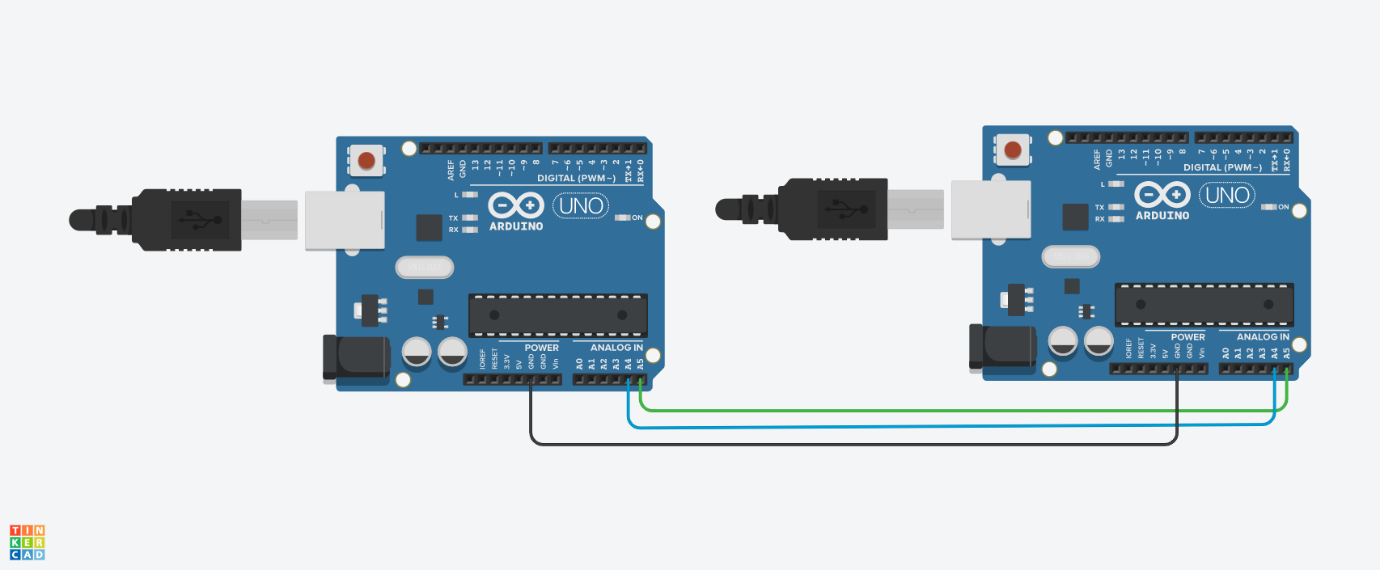
Serial.print(a)

}

void loop()

{

}



**3)**

//Client

#include<Ethernet.h>

#include<SPI.h>

byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED};

byte ip[]= {10, 0, 0, 177};

byte dns[] ={64, 233, 187, 99};

EthernetClient client; //object

void setup()

{

Ethernet.begin(mac,ip);

Serial.begin(9600);

if(client.connect(serveraddr, 80))

{

    Serial.println(“successful”);

}

else

{

    Serial.println(“fail”);

}

}

void loop()

{

if(client.available())

{

char c = client.read();

Serial.println(c);

}

else

{

if(!client.connected())

{

Serial.println(“stopping”)

client.stop();

}

}

}

//server

#include<Ethernet.h>

#include<SPI.h>

EthernetServer server(80);

byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED};

IPaddress ip(192,168,23,8);

void setup()

{

Serial.begin(9600);

Ethernet.begin(mac,ip);

server.begin();

Serial.println(Ethernet.localIP());

}

byte sensorreading = 0;

void loop()

{

EthernetClient client = server.available();

if(client)

{

while(client.connected())

{

client.write(x);

}

client.stop();

Serial.println(“disconnect client”);

}

}

**4)**

#include<SPI.h>

#include<WiFi.h>

char ssid[] = {“wifiname”};

void setup()

{

Serial.begin(9600);

if(WiFi.status()==WL\_NO\_SHIELD)

{

Serial.print(“wifi shield not present”);

}

while(WiFi.status()!=WL\_CONNECTED)

{

WiFi.begin(ssid);

delay(10000);

}

Serial.println(“Wifi connected”);

}

void loop()

{

IPaddress ip = WiFi.localIP()

Serial.println(ip);

byte mac[6];

WiFi.macAddress(mac)

Serial.print(mac[5], HEX);

Serial.print(“:”);

Serial.print(mac[4], HEX);

Serial.print(“:”);

Serial.print(mac[3], HEX);

Serial.print(“:”);

Serial.print(mac[2], HEX);

Serial.print(“:”);

Serial.print(mac[1], HEX);

Serial.print(“:”);

Serial.println(mac[0], HEX);

IPaddress subnet = WiFi.subnetMask();

Serial.println(subnet);

IPaddress gateway= WiFi.gatewayIP();

Serial.println(gateway);

Serial.println(WiFi.SSID());

long strength = WiFi.RSSI();

Serial.println(strength);

byte routerip[6];

WiFi.BSSID(routerip)

Serial.print(routerip[5], HEX);

Serial.print(“:”);

Serial.print(routerip[4], HEX);

Serial.print(“:”);

Serial.print(routerip[3], HEX);

Serial.print(“:”);

Serial.print(routerip[2], HEX);

Serial.print(“:”);

Serial.print(routerip[1], HEX);

Serial.print(“:”);

Serial.println(routerip[0], HEX);

byte encryption = WiFi.encryptionType();

Serial.println(encryption, HEX);

}