Design Document

CSE658: Seminar on Mobile and Wireless Networking

Pratik Musale, Kyeong Joo Jung, David (Bong Jun Choi)

Design Document

Contents

List of Figures

1. Objective

The objective of the mobile and wireless networking seminar was to have a prototype of quadcopter sensors application. To do this, we went through various applications and sensors. We were focusing on the application on how the quadcopters can align itself in the mesh network. Therefore, there was a need for a continuous sending and receiving communications of critical data among the quadcopters.

2. Component Selection

Quadcopters are one of the IoT devices. IoT devices these days cannot use high used battery and lots of computational problems. Hence, we needed to select components which has low requirements in battery and small computation power. To satisfy the requirements, we selected Arduino Rev 3 board.

Arduino Rev 3 board utilizes a 9V ~ 12V DC supply voltage with operating voltage of 5V. Additionally, it also provides two DC voltage sources of 5V and 3.3V. Above all, Arduino Rev 3 has ATmega328P controller, which provides 2KB of SRAM, 1KB of EPROM and 32KB of flash memory. In addition, the controller also provides 14 Digital (6 PWM including) and 6 Analog Input / Output pins. Since the controller has large I/O options, we can select wide variety of sensors.

We selected two components:

i. Ultrasonic PING Sensor: 1EA
ii. NRF24L01 (Transmitter: 1EA / Receiver: 1EA)

3. Sensors and Library Functions

i. Ultrasonic PING Sensor: This sensor utilizes 5V DC as input voltage which can be provided by Arduino Board. Ultrasonic sensor requires 3 Digital Pins, one for Signalling, one for Input Voltage and the last one for Ground. Signalling pin works both for transmitter and receiver. To ensure a clean HIGH pulse of ultrasonic waves the signalling pin is first given a 2 microsecond HIGH pulse followed by a LOW Pulse and lastly a high duration HIGH Pulse to measure distance. The distance is measured on the principal of time taken by the waves from transmitter to the obstacle and the echo back from the obstacle to the sensor's receiver.

ii. NRF24L01 as Transmitter: The two main connections required here besides the power and ground are the CE (Chip Enable required to select chip for Transmitter and Receiver) and CSN (SPI Chip Select).

   We use the special library of NRF24L01. The library provides radio (CE, CSN), which gives us payload size of 32 bytes, dynamic payload disabled and address width of 5 bytes. We set the power amplifier PA Level at minimum because the transmitter and receiver is very close.

iii. NRF24L01 as Receiver: The initial setup will be same as the transmitter, but we need to set the baud rate beforehand for the receiver. By default, the transmitter uses 9600 baud rate for transmitting.

   In addition, to make both transmitter and receiver communicate, we need to set any address in 5 numbers to make sure that communication between the designated ones are operating.

4. Circuit Diagram

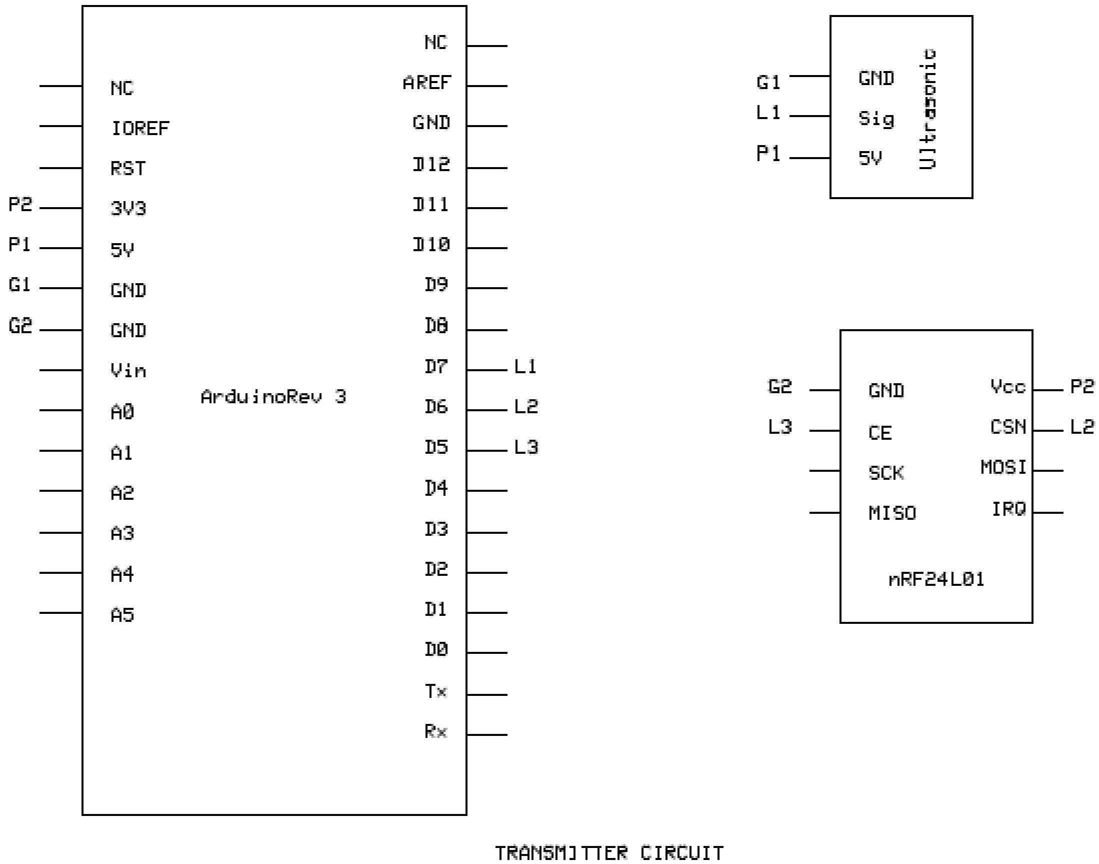

Figure 1: Transmitter Circuit

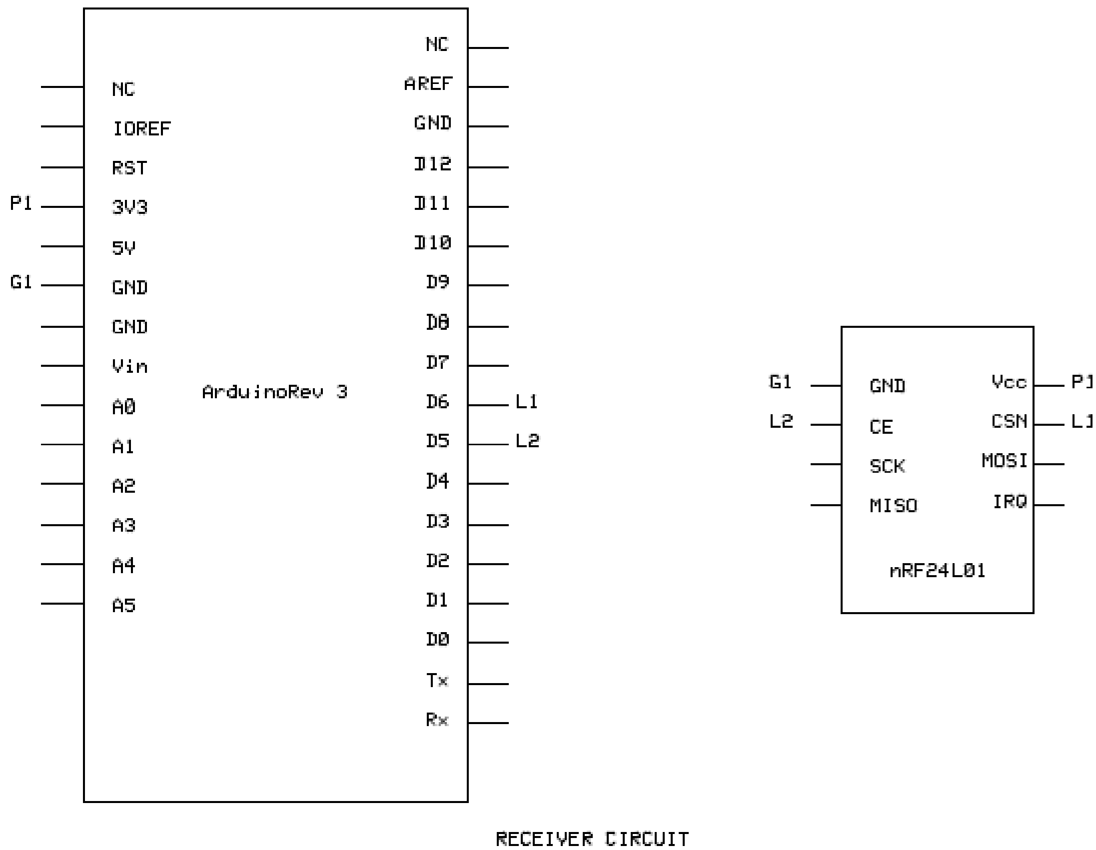

Figure 2: Receiver Circuit

Note: The transmitter circuit first measures the distance and then sends to the main receiver.

5. Source Code

1. Transmitter Code

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
RF24 radio(7, 8); // CE, CSN
const byte address[6] = "00001"; //address to communicate with receiver

const int pingPin=9; //this pin number is for the ultrasonic sensor signal

void setup(){
  Serial.begin(9600);

  radio.begin();
  radio.openWritingPipe(address); //initialize the radio object and set the address of the receiver
  radio.setPALevel(RF24_PA_MIN); //This is to set the Power Amplifier level
  radio.stopListening(); //for transmitter
}

void loop(){
  long duration,cm; //establishing variables for duration of the ping and the distance(cm)
  pinMode(pingPin, OUTPUT);
  digitalWrite(pingPin,LOW);
  delayMicroseconds(2);
  digitalWrite(pingPin,HIGH);
  delayMicroseconds(5);
  digitalWrite(pingPin,LOW);//Ping will be triggered by a HighPulse of 2 or more microsec.
                          //Then give a short LOW pulse beforehand to ensure a clean High pulse.
  pinMode(pingPin, INPUT); //The same pin will be used to read the signal from PING. high pulse duration is the
time from the sending of the ping to the reception of its echo off of an object.
  duration=pulseIn(pingPin, HIGH);


  cm=microsecondsToCentimeters(duration);//Converting the time into distance

  radio.write(&cm, sizeof(cm)); //this function will send the value of cm to the recevier

  delay(100);
}

long microsecondsToCentimeters(long microseconds){
  return microseconds / 29 / 2; //speed of sound is 340 m/s or 29 microsec per cm. So following is the fomula to
get the distance using the time
}
```

Transmitter Code Drive Link

2. Receiver Code

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
RF24 radio(7, 8); // CE, CSN
const byte address[6] = "00001"; //address to communicate with the transmitter
void setup() {
  Serial.begin(9600);
  radio.begin();
  radio.openReadingPipe(0, address); //set the address so it can start to communicate between 2 modules
  radio.setPALevel(RF24_PA_MIN);//This is to set the Power Amplifier level
  radio.startListening(); //for receiver
}
void loop() {
  if (radio.available()) { //checking whether there is a data to be received
    long text;
    radio.read(&text, sizeof(text)); //it will save the incoming data
    Serial.println(text);
  }
}
```

Receiver Code Drive Link