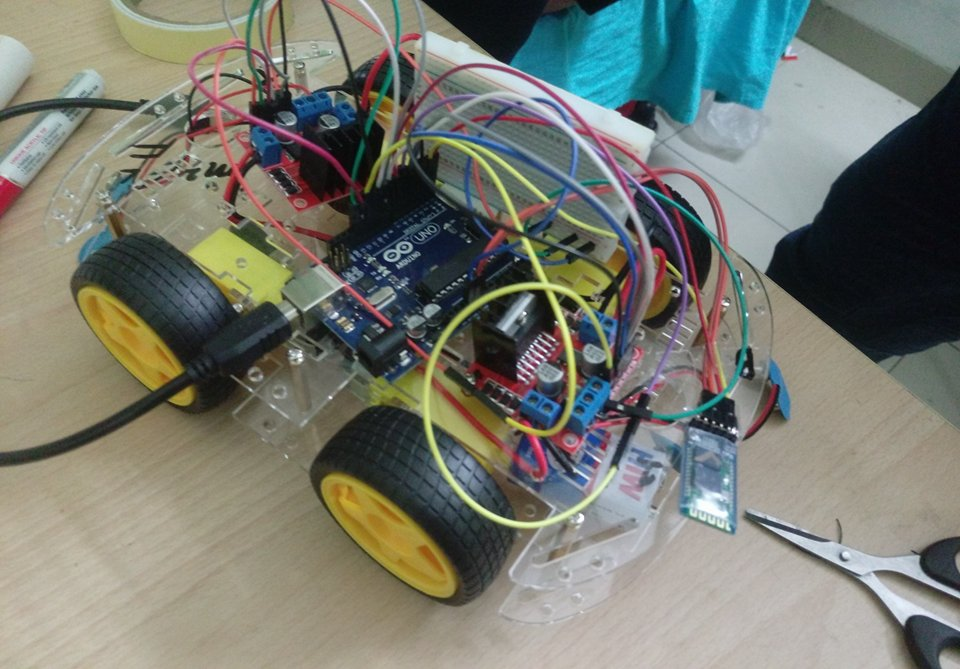
**Spykar**

(Bluetooth based surveillance car using Arduino UNO)

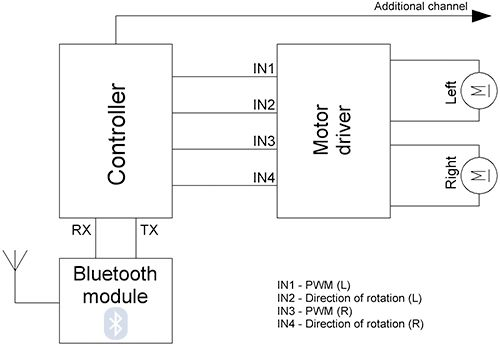


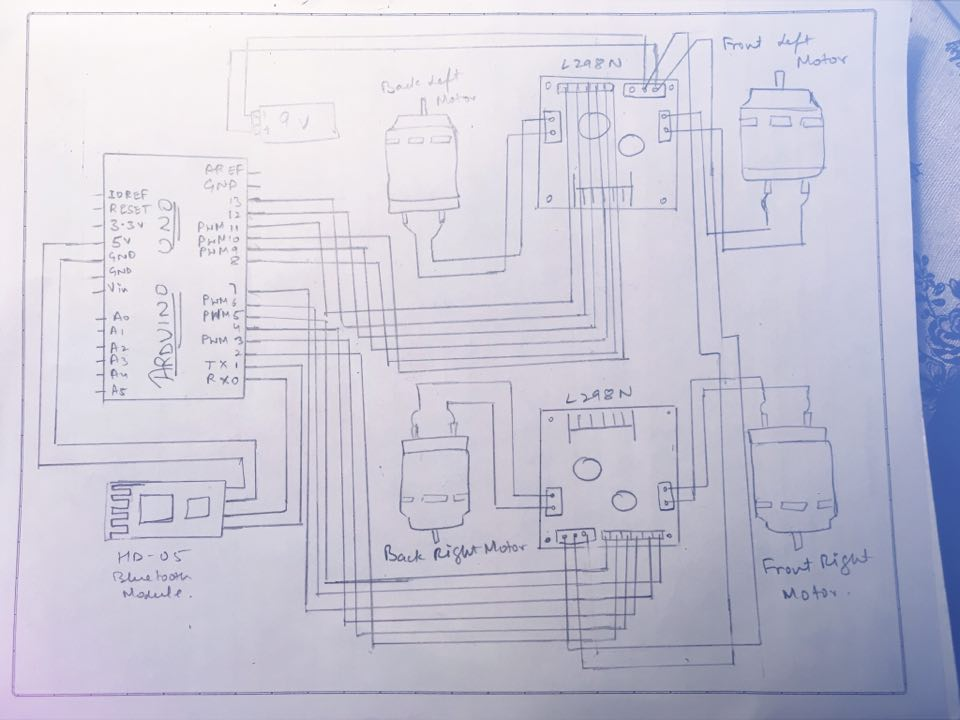
|  |  |  |
| --- | --- | --- |
| Name | Section | Enrollment No. |
| Nandini Goyal | S3 | U101116FCS074 |
| Rahul Saha | S3 | U101116FCS241 |
| Aman Khandelwal | S1 | U101116FCS007 |
| Ajay Sharma | S1 | U101116FCS006 |

**Abstract:**

Spykar is a Bluetooth based surveillance car using Arduino UNO. It works on the basis of tilt control using an android based app called 'Bluetooth RC Controller' via an android smartphone. This car would have another phone which would act as a camera for the car which is connected to a laptop via wifi. There on the computer one can see whatever the spykar records.

**Block diagram:**





**What is surveillance system?**

Surveillance is the monitoring of [behavior](https://en.wikipedia.org/wiki/Behavior), activities, or other changing information for the purpose of influencing, managing, directing, or protecting people. This can include observation from a distance by means of electronic equipment (such as [closed-circuit television](https://en.wikipedia.org/wiki/Closed-circuit_television) (CCTV) cameras) or interception of electronically transmitted information (such as [Internet traffic](https://en.wikipedia.org/wiki/Internet_traffic) or phone calls). It can also include simple no- or relatively low-technology methods such as human intelligence agents and [postal interception](https://en.wikipedia.org/wiki/Postal_interception).

**Components used:**

1) Arduino UNO

2) HC-05 Bluetooth module

3) 4-wheel Drive Robot Smart Car Chassis Kit

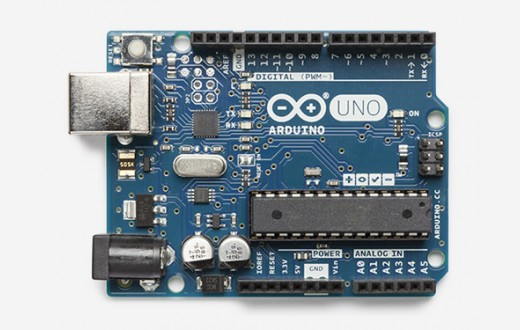
4) IC-Motor driver- L298N

5) Android app ( Bluetooth RC Controller)

6) Connecting Jumper Wires

7) 9V Batteries (4)

**Arduino UNO**



The Arduino Uno is a microcontroller board based on the ATmega328 ([datasheet](http://www.atmel.com/dyn/resources/prod_documents/doc8161.pdf)). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

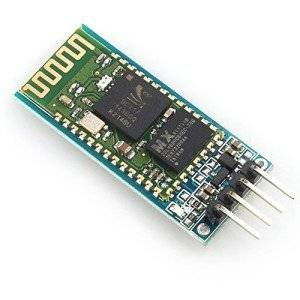
The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode.

Revision 3 of the board has the following new features:

* 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes.
* Stronger RESET circuit.
* Atmega 16U2 replace the 8U2.

**HC-05 Bluetooth Module:**

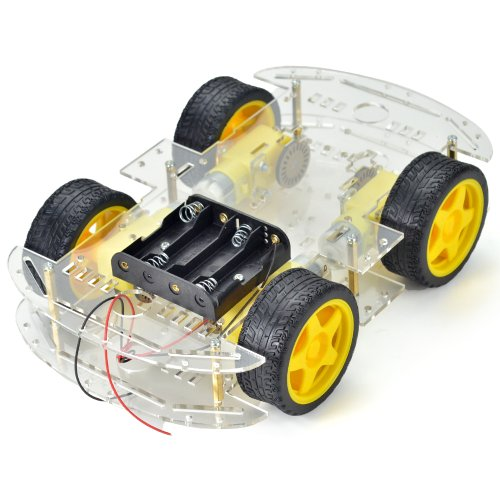


This is HC-05 wireless bluetooth RF transceiver module for Arduino CSR Bluetooth Chip Solution Bluetooth Spec v2.0+EDR Compliant Enhanced Data Rate (EDR) compliant with V2.0.E.2 of specification for both 2Mbps and 3Mbps modulation modes Full Speed Bluetooth Operation with Full Piconet Support and Scatternet Support Incredible small size with 3.3V input, and RoHS Compliant UART interface and with baudrate setup function Support for 8Mbit External Flash Onboard Support for 802.11Co-Existence

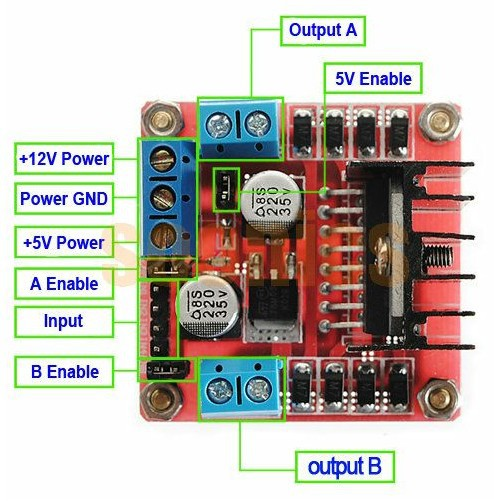
Specifications:

Bluetooth protocol:　Bluetooth Specification v2.0+EDR USB protocol: USB v1.1/2.0 Frequency: 2.4GHz ISM band Modulation: GFSK(Gaussian Frequency Shift Keying) Transmit power: ≤4dBm, Class 2 Sensitivity: ≤-84dBm at 0.1% BER Rate: Asynchronous: 2.1Mbps(Max) / 160 kbps Synchronous: 1Mbps/1Mbps Security features: Authentication and encryption Support profiles: Bluetooth serial port (master & slave) Power Supply: +3.3VDC 50mA Working temperature: –5 ~ +45 Centigrade.

**4-wheel Drive Robot Smart Car Chassis Kit**



**IC-Motor Driver L298N:**



Double H driver module uses ST L298N dual full-bridge driver, an integrated monolithic circuit in a 15- lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the con-nection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

**Connecting Jumper wires:**



9V Batteries:



**Code:**

#define ENA\_m1 5 // Enable/speed motor Front Right

#define ENB\_m1 6 // Enable/speed motor Back Right

#define ENA\_m2 10 // Enable/speed motor Front Left

#define ENB\_m2 11 // Enable/speed motor Back Left

#define IN\_11 2 // L298N #1 in 1 motor Front Right

#define IN\_12 3 // L298N #1 in 2 motor Front Right

#define IN\_13 4 // L298N #1 in 3 motor Back Right

#define IN\_14 7 // L298N #1 in 4 motor Back Right

#define IN\_21 8 // L298N #2 in 1 motor Front Left

#define IN\_22 9 // L298N #2 in 2 motor Front Left

#define IN\_23 12 // L298N #2 in 3 motor Back Left

#define IN\_24 13 // L298N #2 in 4 motor Back Left

int command; //Int to store app command state.

int speedCar = 100; // 50 - 255.

int speed\_Coeff = 4;

void setup() {

pinMode(ENA\_m1, OUTPUT);

pinMode(ENB\_m1, OUTPUT);

pinMode(ENA\_m2, OUTPUT);

pinMode(ENB\_m2, OUTPUT);

pinMode(IN\_11, OUTPUT);

pinMode(IN\_12, OUTPUT);

pinMode(IN\_13, OUTPUT);

pinMode(IN\_14, OUTPUT);

pinMode(IN\_21, OUTPUT);

pinMode(IN\_22, OUTPUT);

pinMode(IN\_23, OUTPUT);

pinMode(IN\_24, OUTPUT);

Serial.begin(9600);

}

void goAhead(){

digitalWrite(IN\_11, HIGH);

digitalWrite(IN\_12, LOW);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, LOW);

digitalWrite(IN\_14, HIGH);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, LOW);

digitalWrite(IN\_22, HIGH);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, HIGH);

digitalWrite(IN\_24, LOW);

analogWrite(ENB\_m2, speedCar);

}

void goBack(){

digitalWrite(IN\_11, LOW);

digitalWrite(IN\_12, HIGH);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, HIGH);

digitalWrite(IN\_14, LOW);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, HIGH);

digitalWrite(IN\_22, LOW);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, LOW);

digitalWrite(IN\_24, HIGH);

analogWrite(ENB\_m2, speedCar);

}

void goRight(){

digitalWrite(IN\_11, LOW);

digitalWrite(IN\_12, HIGH);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, HIGH);

digitalWrite(IN\_14, LOW);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, LOW);

digitalWrite(IN\_22, HIGH);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, HIGH);

digitalWrite(IN\_24, LOW);

analogWrite(ENB\_m2, speedCar);

}

void goLeft(){

digitalWrite(IN\_11, HIGH);

digitalWrite(IN\_12, LOW);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, LOW);

digitalWrite(IN\_14, HIGH);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, HIGH);

digitalWrite(IN\_22, LOW);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, LOW);

digitalWrite(IN\_24, HIGH);

analogWrite(ENB\_m2, speedCar);

}

void goAheadRight(){

digitalWrite(IN\_11, HIGH);

digitalWrite(IN\_12, LOW);

analogWrite(ENA\_m1, speedCar/speed\_Coeff);

digitalWrite(IN\_13, LOW);

digitalWrite(IN\_14, HIGH);

analogWrite(ENB\_m1, speedCar/speed\_Coeff);

digitalWrite(IN\_21, LOW);

digitalWrite(IN\_22, HIGH);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, HIGH);

digitalWrite(IN\_24, LOW);

analogWrite(ENB\_m2, speedCar);

}

void goAheadLeft(){

digitalWrite(IN\_11, HIGH);

digitalWrite(IN\_12, LOW);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, LOW);

digitalWrite(IN\_14, HIGH);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, LOW);

digitalWrite(IN\_22, HIGH);

analogWrite(ENA\_m2, speedCar/speed\_Coeff);

digitalWrite(IN\_23, HIGH);

digitalWrite(IN\_24, LOW);

analogWrite(ENB\_m2, speedCar/speed\_Coeff);

}

void goBackRight(){

digitalWrite(IN\_11, LOW);

digitalWrite(IN\_12, HIGH);

analogWrite(ENA\_m1, speedCar/speed\_Coeff);

digitalWrite(IN\_13, HIGH);

digitalWrite(IN\_14, LOW);

analogWrite(ENB\_m1, speedCar/speed\_Coeff);

digitalWrite(IN\_21, HIGH);

digitalWrite(IN\_22, LOW);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, LOW);

digitalWrite(IN\_24, HIGH);

analogWrite(ENB\_m2, speedCar);

}

void goBackLeft(){

digitalWrite(IN\_11, LOW);

digitalWrite(IN\_12, HIGH);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, HIGH);

digitalWrite(IN\_14, LOW);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, HIGH);

digitalWrite(IN\_22, LOW);

analogWrite(ENA\_m2, speedCar/speed\_Coeff);

digitalWrite(IN\_23, LOW);

digitalWrite(IN\_24, HIGH);

analogWrite(ENB\_m2, speedCar/speed\_Coeff);

}

void stopRobot(){

digitalWrite(IN\_11, LOW);

digitalWrite(IN\_12, LOW);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, LOW);

digitalWrite(IN\_14, LOW);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, LOW);

digitalWrite(IN\_22, LOW);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, LOW);

digitalWrite(IN\_24, LOW);

analogWrite(ENB\_m2, speedCar);

}

void loop(){

if (Serial.available() > 0) {

command = Serial.read();

stopRobot(); //Initialize with motors stopped.

switch (command) {

case 'F':goAhead();break;

case 'B':goBack();break;

case 'L':goLeft();break;

case 'R':goRight();break;

case 'D':stopRobot();break;

case 'I':goAheadRight();break;

case 'G':goAheadLeft();break;

case 'J':goBackRight();break;

case 'H':goBackLeft();break;

case '0':speedCar = 100;break;

case '1':speedCar = 115;break;

case '2':speedCar = 130;break;

case '3':speedCar = 145;break;

case '4':speedCar = 160;break;

case '5':speedCar = 175;break;

case '6':speedCar = 190;break;

case '7':speedCar = 205;break;

case '8':speedCar = 220;break;

case '9':speedCar = 235;break;

case 'q':speedCar = 255;break;

}

}

}