Drive motor and read encoder-USART

1.1 Explanation

Please read 《0. Motor introduction and usage》first to understand the motor parameters, wiring method, and power supply voltage you are currently using. To avoid improper operation and damage to the driver board or motor.

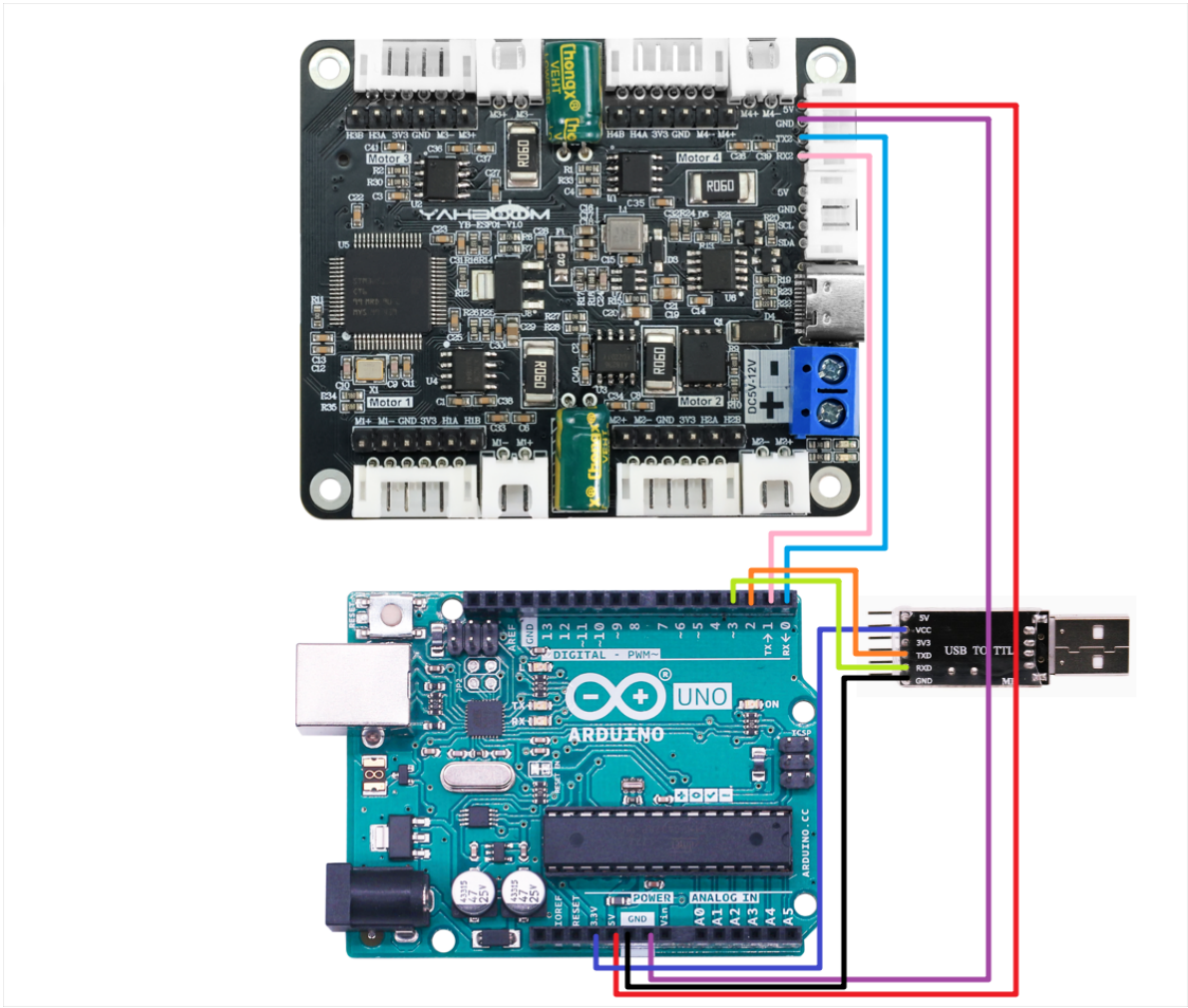
I2C and serial communication cannot be shared, only one can be selected.

The course uses the Arduino UNO board. And use is Arduino 1.8.5 IDE

Before writing the program, do not connect the driver board to the RX pin on the Arduino, otherwise the program cannot be written into the board.

After the program is written, connect the RX pin on the Arduino.

Hardware wiring:



Motor	4-channel motor drive board(Motor)
M2	M-
V	3V3
A	H1A
B	H1B
G	GND
M1	M+

4-channel motor drive board	Arduino UNO
RX2	TX
TX2	RX
GND	GND
5V	5V

Since the hardware serial port on the Arduino board is used to communicate with the driver board, this case requires an additional USB to TTL serial port to print data.

USB TO TTL	Arduino UNO
VCC	3V3
GND	GND
RXD	3
TXD	2

Serial port configuration: **Baud rate 115200, no parity check, no hardware flow control, 1 stop bit**

1.2 Code analysis

```
#define UPLOAD_DATA 3  //0:不接受数据 1:接收总的编码器数据 2:接收实时的编码器 3:接收电机当前速度 mm/s
                       //0: Do not receive data 1: Receive total encoder data 2: Receive real-time encoder 3: Receive current motor speed mm/s

#define MOTOR_TYPE 1   //1:520电机 2:310电机 3:测速码盘TT电机 4:TT直流减速电机 5:L型520电机
                       //1:520 motor 2:310 motor 3:speed code disc TT motor 4:TT DC reduction motor 5:L type 520 motor
```

- UPLOAD_DATA: used to set the data of the motor encoder. Set 1 to the total number of encoder pulses and 2 to the real-time pulse data of 10ms.

- MOTOR_TYPE: used to set the type of motor used. Just modify the corresponding numbers according to the comments according to the motor you are currently using. You don't need to modify the rest of the code.

If you need to drive the motor and observe the data, just modify the two numbers at the beginning of the program. No changes are required to the rest of the code.

```
  #if MOTOR_TYPE == 1
    send_motor_type(1);//配置电机类型 Configure motor type
    delay(100);
    send_pulse_phase(30);//配置减速比 查电机手册得出    Configure the reduction
ratio. Check the motor manual to find out
    delay(100);
    send_pulse_line(11);//配置磁环线 查电机手册得出 Configure the magnetic ring wire.
Check the motor manual to get the result.
    delay(100);
    send_wheel_diameter(67.00);//配置轮子直径,测量得出        Configure the wheel
diameter and measure it
    delay(100);
    send_motor_deadzone(1900);//配置电机死区,实验得出 Configure the motor dead zone,
and the experiment shows
    delay(100);

  #elif MOTOR_TYPE == 2
  send_motor_type(2);
    delay(100);
    send_pulse_phase(20);
    delay(100);
    send_pulse_line(13);
    delay(100);
    send_wheel_diameter(48.00);
    delay(100);
    send_motor_deadzone(1600);
    delay(100);

  #elif MOTOR_TYPE == 3
  send_motor_type(3);
    delay(100);
    send_pulse_phase(45);
    delay(100);
    send_pulse_line(13);
    delay(100);
    send_wheel_diameter(68.00);
    delay(100);
    send_motor_deadzone(1250);
    delay(100);

  #elif MOTOR_TYPE == 4
  send_motor_type(4);
    delay(100);
    send_pulse_phase(48);
    delay(100);
    send_motor_deadzone(1000);
    delay(100);
```

```
  #elif MOTOR_TYPE == 5
  send_motor_type(1);
    delay(100);
    send_pulse_phase(40);
    delay(100);
    send_pulse_line(11);
    delay(100);
    send_wheel_diameter(67.00);
    delay(100);
    send_motor_deadzone(1900);
    delay(100);
  #endif
```

This is used to store the parameters of the Yahboom motor. By modifying the MOTOR_TYPE parameter above, one-click configuration can be achieved.

In normally, do not modify the code here when using the Yahboom motor.

If you are using your own motor, or if a certain data needs to be modified according to your needs, you can check the course《1.2 Control command》 to understand the specific meaning of each command.

```
void loop(){

      Motor_USART_Recieve();
      if(g_recv_flag == 1)
    {
      g_recv_flag = 0;
      #if MOTOR_TYPE == 4
        Contrl_Pwm(i*2,i*2,i*2,i*2);
      #else
        Contrl_Speed(i,i,i,i);   //值为-1000~1000 The value is -1000~1000
      #endif

        Deal_data_real();
        //delay(100);
      #if UPLOAD_DATA == 1

 sprintf(buffer,"M1:%d,M2:%d,M3:%d,M4:%d\r\n",Encoder_Now[0],Encoder_Now[1],Enco
der_Now[2],Encoder_Now[3]);
        printSerial.println(buffer);
      #elif UPLOAD_DATA == 2

 sprintf(buffer,"M1:%d,M2:%d,M3:%d,M4:%d\r\n",Encoder_Offset[0],Encoder_Offset[1
],Encoder_Offset[2],Encoder_Offset[3]);
        printSerial.println(buffer);
      #elif UPLOAD_DATA == 3
        dtostrf(g_Speed[0], 4, 2, string1);
        dtostrf(g_Speed[1], 4, 2, string2);
        dtostrf(g_Speed[2], 4, 2, string3);
        dtostrf(g_Speed[3], 4, 2, string4);

 sprintf(buffer,"M1:%s,M2:%s,M3:%s,M4:%s\r\n",string1,string2,string3,string4);
        printSerial.println(buffer);
      #endif
```

```
      i++;
      if (i == 1000) i = 0;

    }

}
```

In the loop program, the speed of the four motors will be slowly increased from 0 to 1000. If the motor type is 4, that is, the motor without encoder, the motor's PWM is directly controlled.

At the same time, the data sent by the driver board is read and printed out at the same time.

```
//检验从驱动板发送过来的数据，符合通讯协议的数据则保存下来
//Check the data sent from the driver board, and save the data that meets the
communication protocol
void Deal_Control_Rxtemp(uint8_t rxtemp)
{
    static u8 step = 0;
    static u8 start_flag = 0;

    if(rxtemp == '$' &&     start_flag == 0)
    {
        start_flag = 1;
        memset(g_recv_buff,0,RXBUFF_LEN);//清空数据 Clear data
    }

    else if(start_flag == 1)
    {
            if(rxtemp == '#')
            {
                start_flag = 0;
                step = 0;
                g_recv_flag = 1;
        // 检查前四个字符  Check the first four characters
    if (strncmp("MAll:",(char*)g_recv_buff,5)==0 ||
        strncmp("MTEP:",(char*)g_recv_buff,5)==0 ||
        strncmp("MSPD:",(char*)g_recv_buff,5)==0) {
        if (isValidNumbers((char*)g_recv_buff + 5)) {
                // 如果符合条件，打印数据  If the conditions are met, print the data
                memcpy(g_recv_buff_deal,g_recv_buff,RXBUFF_LEN);
            }
    } else {
        // 不匹配时清除缓冲区，避免残留无效数据 Clear the buffer when there is no match
to avoid residual invalid data
        memset(g_recv_buff, 0, RXBUFF_LEN);
    }
            }
            else
            {
                if(step > RXBUFF_LEN)
                {
                    start_flag = 0;
                    step = 0;
```

```
                    memset(g_recv_buff,0,RXBUFF_LEN);//清空接收数据   Clear
received data
                }
                else
                {
                    g_recv_buff[step] = rxtemp;
                    step++;
                }
            }
    }

}

//将从驱动板保存到的数据进行格式处理，然后准备打印
//Format the data saved from the driver board and prepare it for printing
void Deal_data_real(void)
{
     static uint8_t data[RXBUFF_LEN];
   uint8_t  length = 0;
    //总体的编码器    Overall encoder
     if ((strncmp("MAll",(char*)g_recv_buff_deal,4)==0))
    {
        length = strlen((char*)g_recv_buff_deal)-5;
        for (uint8_t i = 0; i < length; i++)
        {
            data[i] = g_recv_buff_deal[i+5]; //去掉冒号 Remove the colon
        }
                data[length] = '\0';
                char* strArray[10];//指针数组 长度根据分割号定义  char 1字节   char* 4
字节    Pointer array The length is defined by the split number char 1 byte char*
4 bytes
                char mystr_temp[4][10] = {'\0'};
                splitString(strArray,(char*)data, ", ");//以逗号切割 Split by
comma
                for (int i = 0; i < 4; i++)
                {
                        strcpy(mystr_temp[i],strArray[i]);
                        Encoder_Now[i] = atoi(mystr_temp[i]);
                }

        }
        //10ms的实时编码器数据  10ms real-time encoder data
        else if ((strncmp("MTEP",(char*)g_recv_buff_deal,4)==0))
    {
        length = strlen((char*)g_recv_buff_deal)-5;
        for (uint8_t i = 0; i < length; i++)
        {
            data[i] = g_recv_buff_deal[i+5]; //去掉冒号 Remove the colon
        }
                data[length] = '\0';

                char* strArray[10];//指针数组 长度根据分割号定义  char 1字节   char* 4
字节       Pointer array The length is defined by the split number char 1 byte
char* 4 bytes
                char mystr_temp[4][10] = {'\0'};
```

```
                splitString(strArray,(char*)data, ", ");//以逗号切割 Split by
comma
                for (int i = 0; i < 4; i++)
                {
                        strcpy(mystr_temp[i],strArray[i]);
                        Encoder_Offset[i] = atoi(mystr_temp[i]);
                }
        }
        //速度    Speed
        else if ((strncmp("MSPD",(char*)g_recv_buff_deal,4)==0))
    {
        length = strlen((char*)g_recv_buff_deal)-5;
        for (uint8_t i = 0; i < length; i++)
        {
            data[i] = g_recv_buff_deal[i+5]; //去掉冒号 Remove the colon
        }
                data[length] = '\0';

                char* strArray[10];//指针数组 长度根据分割号定义  char 1字节   char* 4
字节       Pointer array The length is defined by the split number char 1 byte
char* 4 bytes
                char mystr_temp[4][10] = {'\0'};
                splitString(strArray,(char*)data, ", ");//以逗号切割 Split by
comma
                for (int i = 0; i < 4; i++)
                {
                        strcpy(mystr_temp[i],strArray[i]);
                        g_Speed[i] = atof(mystr_temp[i]);
                }
        }
}
```

- Deal_Control_Rxtemp: Filter the received data and save those that meet the communication protocol.
- Deal_data_real: Extract the saved original data and reconstruct a new print format.

1.3 Experimental phenomenon

Before writing the program, do not connect the driver board to the RX pin on the Arduino, otherwise the program cannot be written into the board.

After the program is written, connect the RX pin on the Arduino.

Connect the USB to TTL module to the computer, use the computer's serial port debugging assistant "UartAssist", open the USB to TTL serial port, and you can receive the processed data.

If you open the serial port of the Arduino motherboard, you may see that the serial port prints the original data. After powering on again, you can see that the motor will gradually speed up, then stop, and repeat.

At the same time, you can see that the printed motor value is constantly changing in the serial port assistant.

M1:1118,M2:1127,M3:1136,M4:1119

M1:1118,M2:1127,M3:1136,M4:1119

M1:1118,M2:1127,M3:1136,M4:1119

M1:1118,M2:1127,M3:1136,M4:1119

M1:1118,M2:1127,M3:1136,M4:1119

M1:1118,M2:1127,M3:1136,M4:1119

M1:1118,M2:1127,M3:1136,M4:1119

M1:1118,M2:1127,M3:1136,M4:1119

M1:1118,M2:1127,M3:1136,M4:1119

M1:1118,M2:1127,M3:1136,M4:1119

M1:1118,M2:1127,M3:1136,M4:1119

M1:1118,M2:1127,M3:1136,M4:1119