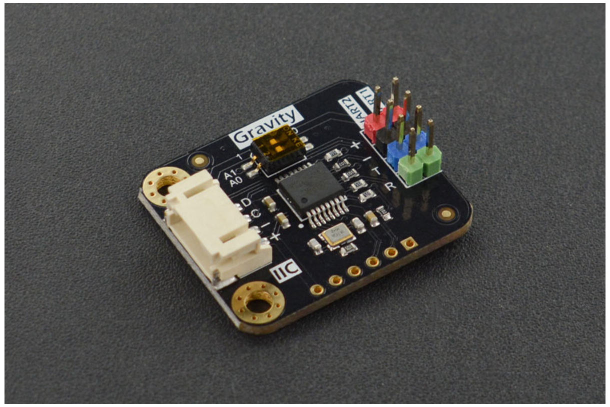


(https://www.dfrobot.com/product-2001.html)

Introduction

This IIC to Dual UART module offers data transmission rate up to 1Mbps, and each sub UART has independent 256-byte FIFO hardware buffer for transmitting and receiving. The baud rate, word length, and check format of every sub UART can be set independently. The module can provide 2Mbps maximum communication rate. At most four such modules can be connected onto one controller board to expand 8 hardware serial ports. This IIC to dual UART module can be extremely suitable for IoT module, Ultrasonic ranging module and GPS module.

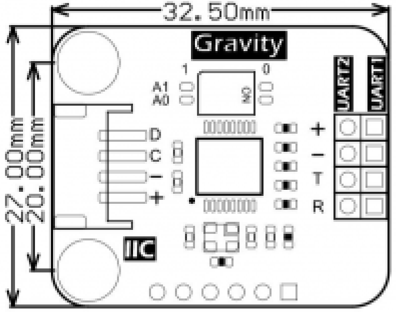
There are usually only 1 or 2 UARTs on Controller boards like Arduino, Raspberry Pi, micro:bit, etc. And one of them must be used in program downloading or debugging. When your application needs to connect several UART devices, you may find that there are not enough UARTs on main-board for connecting. Thus, this module could be a perfect solution for the above situation, or you can use this product in IR receiver module and WS2818 RGB relevant projects with strict timing requirements.

Specification

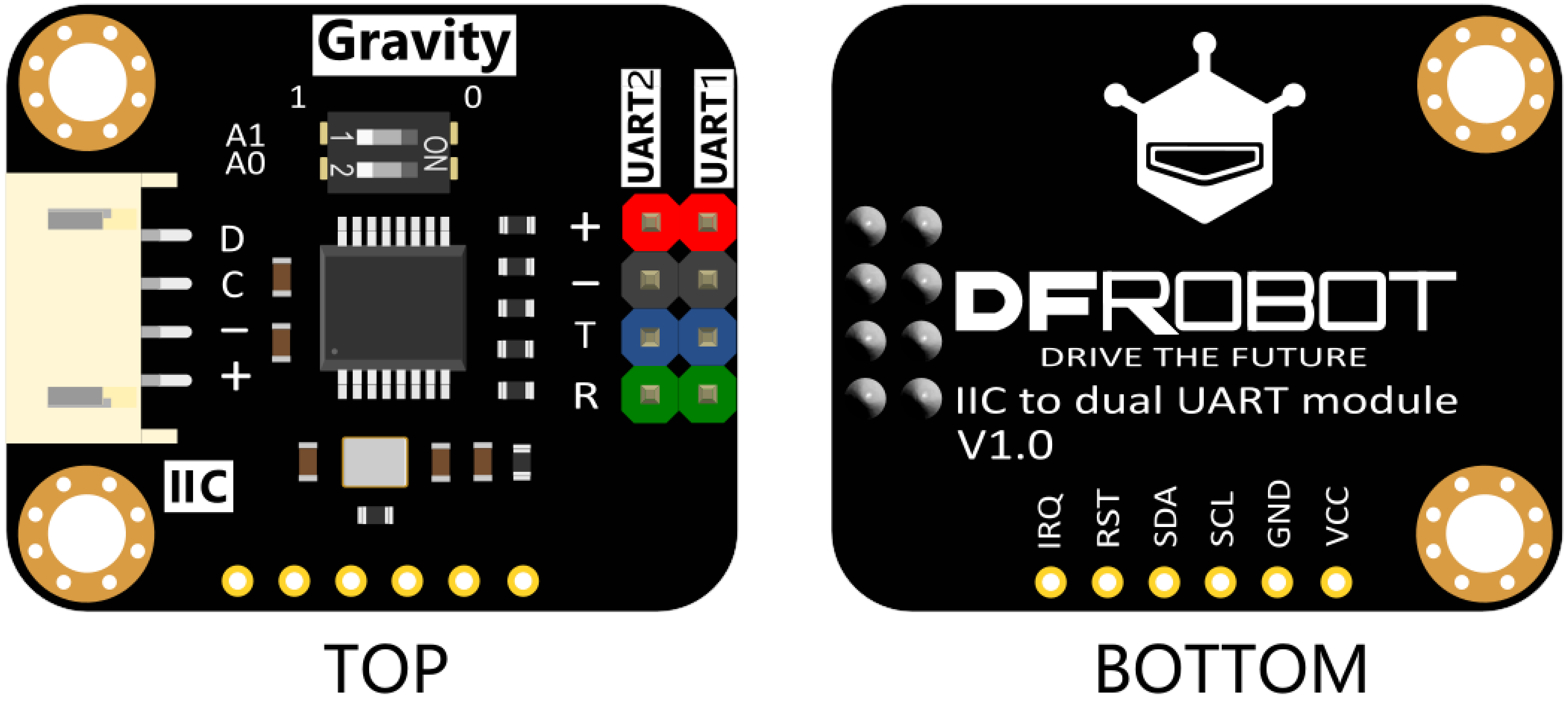
- Operating Voltage: 3.3V-5V

- Operating Current: ‹3mA
- Communication Port: Gravity-IIC 4Pin
- IIC Address: regulated by transmission protocol, refer to tutorial

- 2 Extended UARTs
- Dimension: 32.5×27mm/1.28×1.1 inches
- Mounting Hole Size: 20mm/0.79"
- Operating Temperature: -40°C~85°C



Board OverView



Silkscreen	Description
D	IIC SDA
C	IIC SCL
-	Negative
+	Positive

T	Transmit
R	Receive

Tutorial

DIP vs IIC Address

The IIC to dual UART module is a non-standard IIC device with various addresses. IIC address has 7 bits. When running the IIC scanning program, as long as the upper 4 bits are the same as that of the module, and there will be reponse. So you can scan multiple IIC addresses with same upper 4 bits on one module.

Module IIC Configuration:

Bit	6	5	4	3	2	1	0
Value	IA1	IA0	1	0	C1	C0	0/1

- The 6th bit: IA1 corresponds with the value of DIP switch Al on the module.
- The 5th bit: IA0 corresponds with the value of DIP switch A0 on the module.

The relation between DIP switch and IA1/IA0

DIP Switch		IA1 and IA0	
A1	A0	IA1	IA0
0	0	0	0
0	1	0	1
1	0	1	0

1	1	1	1

- The 4th bit: fixed, 1.

- The 3rd bit: fixed, 0.
- The 2nd and 1st bits: C1/C0 represents the sub UART channel number, for example. 00 for UART1, 01 for UART2.
- The 0 bit: represents the operation object, 0 for register, 1 for FIFO.

Construct Object

When construct IIC to dual UART object, it is required to specify the sub UART(UART1 or UART2) and the value of IA2 and IA0. The constructor is shown below：

```
DFRobot_IIC_Serial(TwoWire &wire = Wire, uint8_t subUartChannel = SUBUART_CHANNEL_1, uint8_t IA1 = 1, uint8_t IA0 = 1);
```

- For operating UART1, pass the parameter SUBUART_CHANNEL_1; for UART2, pass SUBUART_CHANNEL_2;
- If the status of the DIP switch on the module is A1(1),A2(0), then we should pass parameters 1 and 0 to the formal parameter IA1 and IA0.

For example, turn the DIP switch A1 to 1, A0 to 0;

- Construct object, UART1;

  ```
  DFRobot_IIC_Serial iicSerial(Wire, SUBUART_CHANNEL_1, /*IA1=*/1, /*IA0=*/0);//consctruct object UART1
  ```

- Construct object, UART2;

  ```
  DFRobot_IIC_Serial iicSerial(Wire, SUBUART_CHANNEL_2, /*IA1=*/1, /*IA0=*/0);//construct object UART2
  ```

Baud Rate Configuration

Just like other hardware serial ports, the baud rate of the extended UARTs on the module need to be set. For instance, set UART1 to 9600, UART2 to 115200, as shown below:

```
iicSerial1.begin(9600);   //init UART1, set baud rate to 9600
iicSerial2.begin(115200); //init UART2, set baud rate to 115200
```

The IIC to dual UART module adopts 14.7456M crystal oscillator, and it supports the following baud rate:

NO.	1	2	3	4	5	6	7	8
Baud Rate	2400	4800	57600	7200	9600	14400	19200	28800
Baud Rate	38400	76800	115200	153600	230400	460800	307200	921600

Users can directly the above baud rate or define their own baud rate, such as 12800, etc.

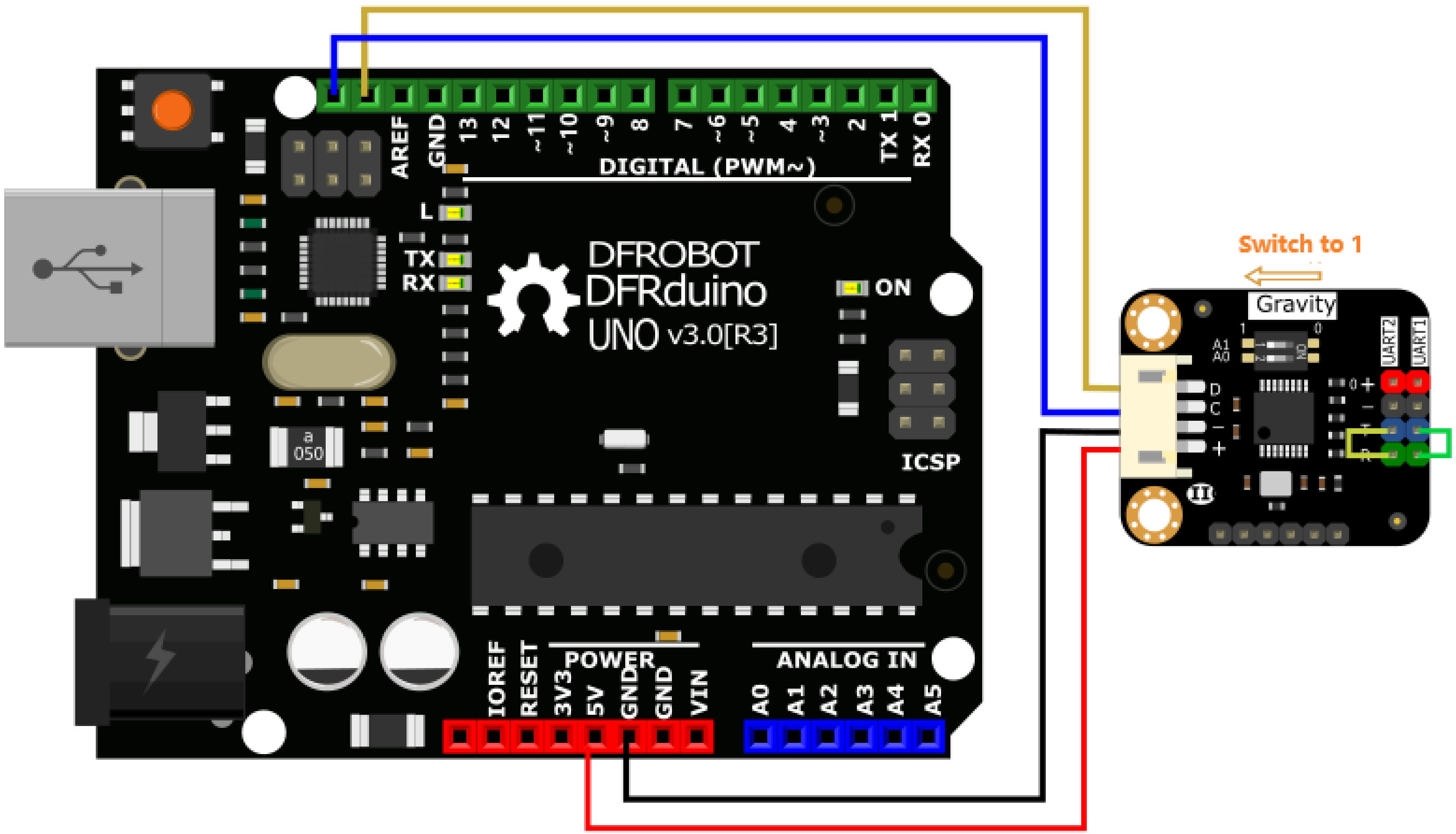
⚠ **NOTE**: Since crystal oscillator and baud rate are closely related, and not all baud rates are supported by the former. So users have to do test on their own when using user-defined baud rate.

Requirements

- **Hardware**
  - DFRduino UNO R3 (https://www.dfrobot.com/product-838.html) (or similar) x 1
  - IIC to dual UART module x1
- **Software**
  - Arduino IDE (https://www.arduino.cc/en/Main/Software)
  - Download and install the **IIC serial Library and Example Program** (https://github.com/DFRobot/DFRobot_IICSerial) (About how to

install the library? (https://www.arduino.cc/en/Guide/Libraries#.UxU8mdzF9H0))

Connection Diagram


Switch to 1

NOTE: Before running the program below, check if all the DIP switchs of the module is turned to "1". If it is not, swtich them to 1, or revise the values passed into the formal parameter IA1 and IA0 of the constrctor in the demo below.

Sample Code 1

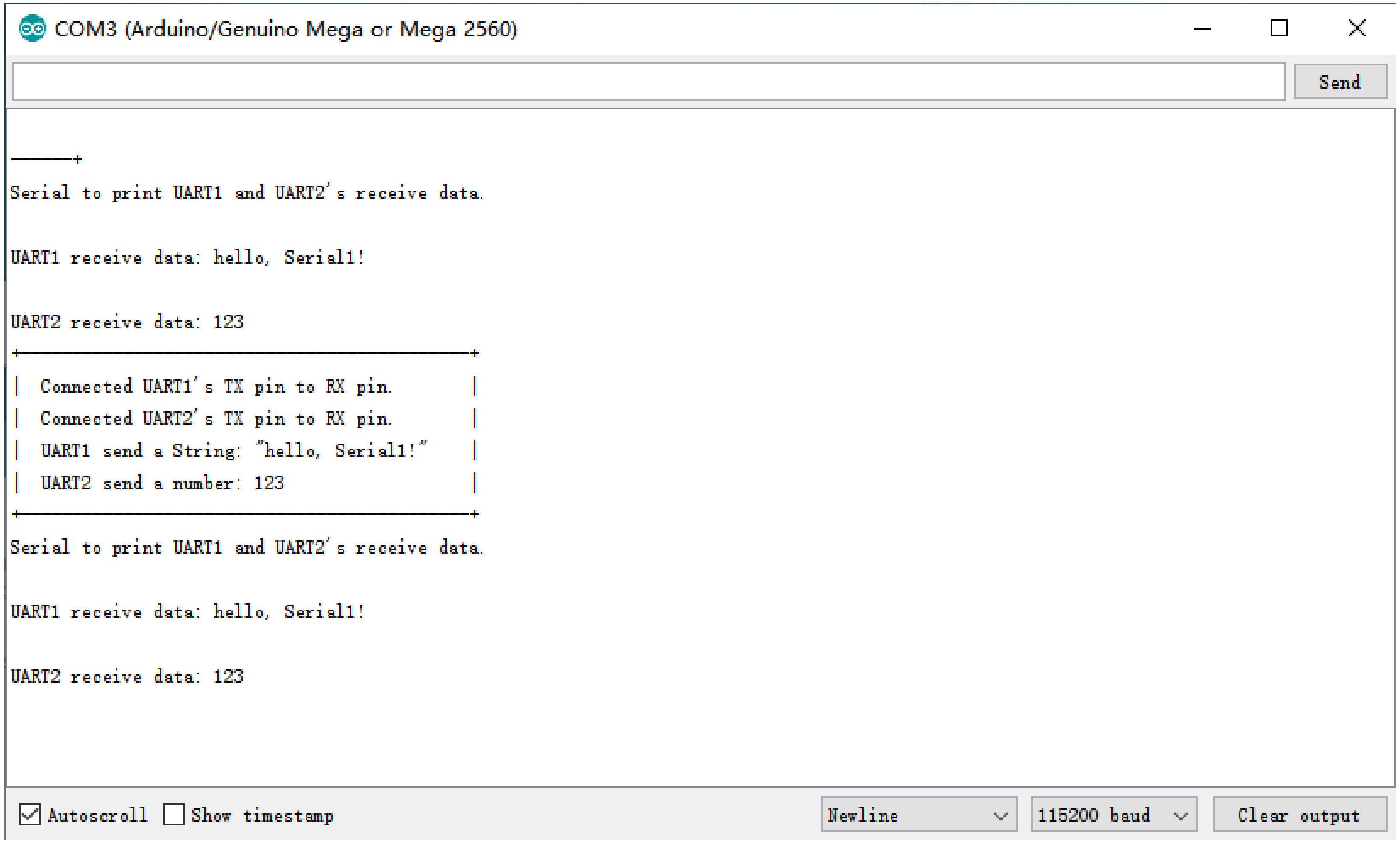
Sample Code 1

Write a demo to allow UART1 and UART2 to realize the function of self-transmitting and self-receiving. Connect pinT of UART1 to its pinR, UART2 TX to its RX. UART1 transmit "hello,Serial1", UART2 transmit "123", then receive their own data and print them out.

```
/*!
 * @file dataTxAndRx.ino
 * @brief Receive and transmit data via UART. Read the data sent by TX pin via pin RX.
 * @n Experiment phenomenon: connect the TX to RX in Sub UART1 and UART2. Read the data sent by Sub UART and print it out.
 *
 * @copyright   Copyright (c) 2010 DFRobot Co.Ltd (https://www.dfrobot.com)
 * @licence     The MIT License (MIT)
 * @author [Arya](xue.peng@dfrobot.com)
 * @version  V1.0
 * @date  2019-07-28
 * @get from https://www.dfrobot.com
 * @url https://github.com/DFRobot/DFRobot_IICSerial
 */
#include <DFRobot_IICSerial.h>
/*DFRobot_IICSerial Constructor
 *Parameter&wire  Wire
 *Parameter subUartChannel sub UART channel, for selecting to operate UART1 or UART2
 *@n Available parameter:
     SUBUART_CHANNEL_1  SUBUART_CHANNEL_2
           UART1             UART2
 *Parameter IA1 corresponds with IA1 Level(0 or 1) of DIP switch on the module, and is used for configuring
 * @n the IIC address of the 6th bit value(default: 1).
 *Parameter IA0 corresponds with IA0 Level(0 or 1) of DIP switch on the module, and is used for configuring
 * @n IIC address of the 5th bit value(default: 1).
 * IIC address configuration:
 * 7   6   5   4   3   2   1   0
 * 0  IA1 IA0  1   0  C1  C0  0/1
 *@n IIC address only has 7 bits, while there are 8 bits for one byte, so the extra one bit will be filled as 0.
 *@n The 6th bit corresponds with IA1 Level of DIP switch, can be configured manually.
 *@n The 5th bit corresponds with IA0 Level of DIP switch, can be configured manually.
 *@n The 4th and 3rd bits are fixed, value 1 and 0 respectively
 *@n The 4th and 3rd bits are fixed, value 1 and 0 respectively
 *@n The values of the 2nd and 1st bits are the sub UART channels, 00 for sub UART 1, 01 for sub UART 2.
 *@n The 0 bit represents the operation object: 0 for register, 1 for FIFO cache.
 */
DFRobot_IICSerial iicSerial1(Wire, /*subUartChannel =*/SUBUART_CHANNEL_1,/*IA1 = */1,/*IA0 = */1);//Construct UART1

//DFRobot_IICSerial iicSerial1;//Default constructor, UART1, IA1 = 1, IA0 = 1
DFRobot_IICSerial iicSerial2(Wire, /*subUartChannel =*/SUBUART_CHANNEL_2, /*IA1 = */1,/*IA0 = */1);//Construct UART2

uint8_t flag = 0;//A flag bit, judge whether to print the prompt information of UART1 and UART2.
//if it is 0, print "UART1 receive data: " or "UART2 receive data: "
void setup() {
  Serial.begin(115200);
  /*begin Init function, set band rate according to the selected crystal frequency.
  begin(long unsigned baud) Call the function, set sub UART band rate.
  default setting->Band rate: baud, data format: IIC_SERIAL_8N1, 8 bits data, no check mode, 1 bit stop bit.
  begin(long unsigned baud, uint8_t format)Use the function to set band rate and data format:
  Parameter supported baud:2400, 4800, 57600, 7200, 9600, 14400, 19200, 28800, 38400,
                76800, 115200, 153600, 230400, 460800, 307200, 921600
  Parameter available format:
  IIC_SERIAL_8N1  IIC_SERIAL_8N2  IIC_SERIAL_8Z1  IIC_SERIAL_8Z2  IIC_SERIAL_8O1
  IIC_SERIAL_8O2  IIC_SERIAL_8E1  IIC_SERIAL_8E2  IIC_SERIAL_8F1  IIC_SERIAL_8F2
  8 represents the number of data bit, N for no parity, Z for 0 parity, O for Odd parity, E for Even parity,
  F for 1 parity, 1 or 2 for the number of stop bit. Default IIC_SERIAL_8N1
  */
  iicSerial1.begin(/*baud = */115200);/*UART1 init*/
  //iicSerial1.begin(/*baud = */115200, /*format = */IIC_SERIAL_8N1);

  iicSerial2.begin(/*baud = */115200);/*UART2 init*/
  Serial.println("\n+--------------------------------------------+");
  Serial.println("|  Connected UART1's TX pin to RX pin.       |");//Connect pin TX and RX of UART1
  Serial.println("|  Connected UART2's TX pin to RX pin.       |");//Connect pin TX and RX of UART2
  Serial.println("|  UART1 send a String: \"hello, Serial1!\"    |");//UART1 transmit a string "hello, Serial1!"
  Serial.println("|  UART2 send a number: 123                  |");//UART2 transmit numbers 123
  Serial.println("+--------------------------------------------+");
  iicSerial1.println("hello, Serial1!");//UART1 transmit string:"hello, Serial1!"
  iicSerial2.write(123);//UART2 transmit:123
  Serial.println("Serial to print UART1 and UART2's receive data.");//print the data received by UART1 and UART2
}

void loop() {
  char c;

  if(iicSerial1.available()){/*available return the number of byte in UART1 receive buffer, none- return 0*/
    flag = 0;
    while(iicSerial1.available()){
      if(flag == 0){
        Serial.print("\nUART1 receive data: ");
        flag = 1;
      }
      c = iicSerial1.read();/*Read data of UART1 receive buffer */
      Serial.print(c);
    }
  }
  if(iicSerial2.available()){
    flag = 0;
    while(iicSerial2.available()){
      if(flag == 0){
        Serial.print("\nUART2 receive data: ");
        flag = 1;
      }
      Serial.print(iicSerial2.read());/*Read and print the data of Sub UART2 receive buffer*/
    }
  }
  delay(1000);
}
```

Expected Result

Serial print the following information.



```
------+
Serial to print UART1 and UART2's receive data.

UART1 receive data: hello, Serial1!

UART2 receive data: 123
+----------------------------------------------+
|  Connected UART1's TX pin to RX pin.         |
|  Connected UART2's TX pin to RX pin.         |
|  UART1 send a String: "hello, Serial1!"      |
|  UART2 send a number: 123                    |
+----------------------------------------------+
Serial to print UART1 and UART2's receive data.

UART1 receive data: hello, Serial1!

UART2 receive data: 123
```

☑ Autoscroll □ Show timestamp Newline 115200 baud

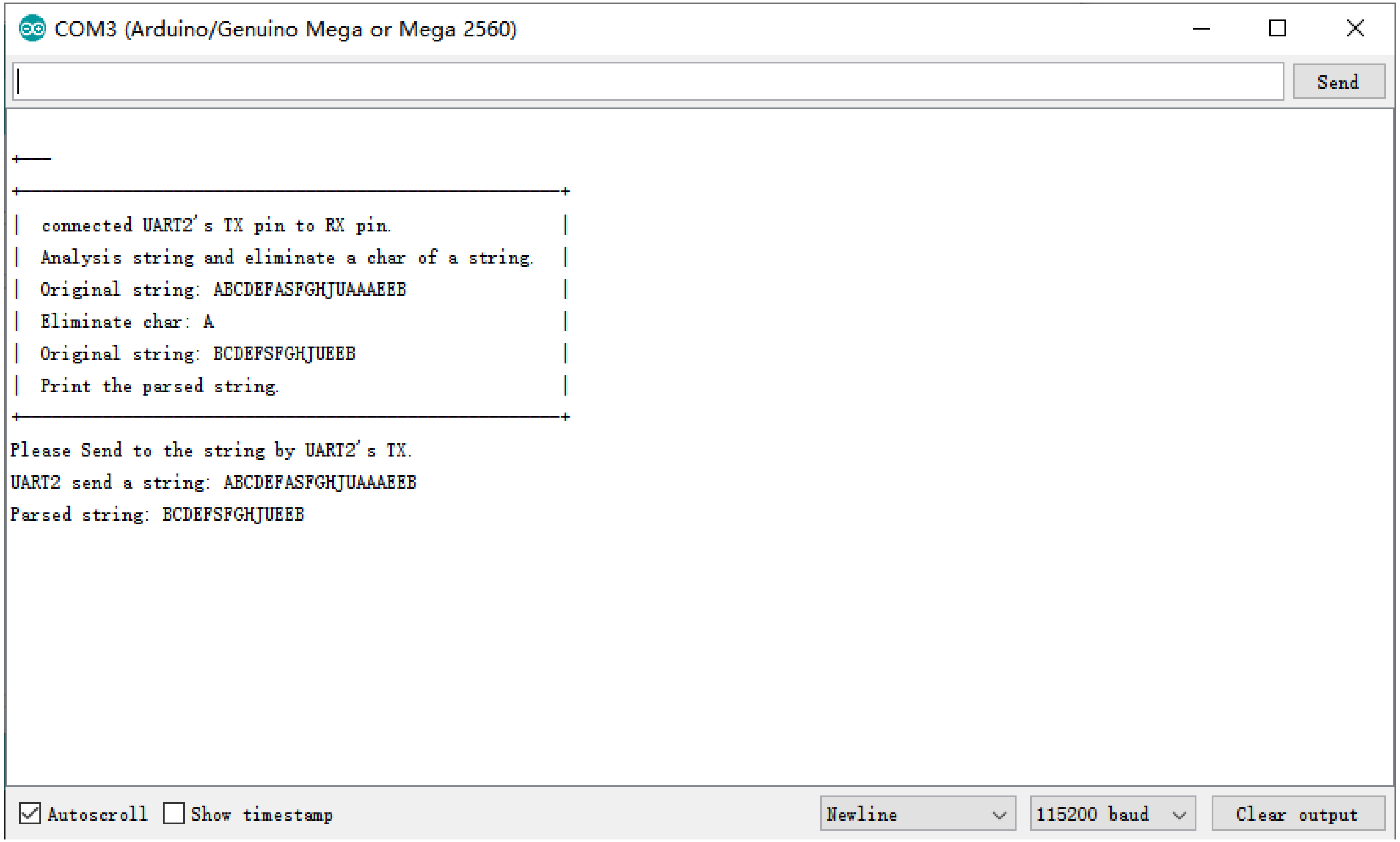
Sample Code 2

Transmit a string "ABCDEFASFGHJUAAAEEB" via UART2. Receive, analyze and revise the string. Delete the "A" in the string and print out the parsed data. (The connection is the same as the image above.)

```
/*!
 * @file cmdAnalysis.ino
 * @brief Analyze UART command, save and print (example: UART2, connect UART2's RX and TX together)
 * @n Transmit a random string via UART: "ABCDEFASFGHJUAAAEEB"
 * @n Receive the string, remove the char "A" of the string, and then print out the new string.
 *
 * @copyright   Copyright (c) 2010 DFRobot Co.Ltd (https://www.dfrobot.com)
 * @licence     The MIT License (MIT)
 * @author [Arya](xue.peng@dfrobot.com)
 * @version  V1.0
 * @date  2019-07-28
 * @get from https://www.dfrobot.com
 * @url https://github.com/DFRobot/DFRobot_IICSerial
 */
#include <DFRobot_IICSerial.h>
/*DFRobot_IICSerial Constructor
 *Parameter &wire  Wire
 *Parameter subUartChannel sub UART channel, for selecting to operate UART1 or UART2
 *@n Available parameter:
     SUBUART_CHANNEL_1  SUBUART_CHANNEL_2
           UART1             UART2
 *Parameter IA1 corresponds with IA1 Level(0 or 1) of DIP switch on the module, and is used for configuring
 * @n the IIC address of the 6th bit value(default: 1).
 *Parameter IA0 corresponds with IA0 Level(0 or 1) of DIP switch on the module, and is used for configuring
 * @n IIC address of the 5th bit value(default: 1).
 * IIC address configuration:
 * 7   6   5   4   3   2   1   0
 * 0  IA1 IA0  1   0  C1  C0  0/1
 *@n IIC address only has 7 bits, while there are 8 bits for one byte, so the extra one bit will be filled as 0.
 *@n The 6th bit corresponds with IA1 Level of DIP switch, can be configured manually.
 *@n The 5th bit corresponds with IA0 Level of DIP switch, can be configured manually.
 *@n The 5th bit corresponds with IA0 Level of DIP switch, can be configured manually.
 *@n The 4th and 3rd bits are fixed, value 1 and 0 respectively
 *@n The values of the 2nd and 1st bits are the sub UART channels, 00 for sub UART 1, 01 for sub UART 2.
 *@n The 0 bit represents the operation object: 0 for register, 1 for FIFO cache.
 */

DFRobot_IICSerial iicSerial2(Wire, /*subUartChannel =*/SUBUART_CHANNEL_2, /*IA1 = */1,/*IA0 = */1);//Construct Sub UART2

char rx_buffer[256];//Define a receive buffer to store the data received by UART2
void setup() {
  Serial.begin(115200);
  /*begin Init function, set band rate according to the selected crystal frequency.
  begin(long unsigned baud) Call the function, set sub UART band rate.
  default setting->Band rate: baud, data format: IIC_SERIAL_8N1, 8 bits data, no check mode, 1 bit stop bit.
  begin(long unsigned baud, uint8_t format) Use the function to set band rate and data format:
  Parameter supported baud: 2400, 4800, 57600, 7200, 9600, 14400, 19200, 28800, 38400,
                76800, 115200, 153600, 230400, 460800, 307200, 921600
  Parameter available format:
  IIC_SERIAL_8N1  IIC_SERIAL_8N2  IIC_SERIAL_8Z1  IIC_SERIAL_8Z2  IIC_SERIAL_8O1
  IIC_SERIAL_8O2  IIC_SERIAL_8E1  IIC_SERIAL_8E2  IIC_SERIAL_8F1  IIC_SERIAL_8F2
  8 represents the number of data bit, N for no parity, Z for 0 parity, O for Odd parity, E for Even parity,
  F for 1 parity, 1 or 2 for the number of stop bit. Default IIC_SERIAL_8N1
  */
  iicSerial2.begin(/*baud = */115200);/*UART2 init*/
  //iicSerial2.begin(/*baud = */115200, /*format = */IIC_SERIAL_8N1);
  Serial.println("\n+-----------------------------------------------------+");
  Serial.println("|  connected UART2's TX pin to RX pin.               |");
  Serial.println("|  Analysis string and eliminate a char of a string.  |");
  Serial.println("|  Original string: ABCDEFASFGHJUAAAEEB               |");
  Serial.println("|  Eliminate char: A                                  |");
  Serial.println("|  Original string: BCDEFSFGHJUEEB                    |");
  Serial.println("|  Print the parsed string.                           |");
  Serial.println("+-----------------------------------------------------+");
  Serial.println("Please Send to the string by UART2's TX.");
  Serial.println("UART2 send a string: ABCDEFASFGHJUAAAEEB");
  iicSerial2.println("ABCDEFASFGHJUAAAEEB");//UART2 transmit string "ABCDEFASFGHJUAAAEEB"
}
```

```
void loop() {
  int n = iicSerial2.available();//Read the number of bytes in UART2
  int i = 0;
  if(n){
      while(iicSerial2.available()){
          if((char)iicSerial2.peek() != 'A'){//Use peek function to read the character without deleting the data in buffer
              rx_buffer[i++] = iicSerial2.read();//Use read function to read character and delete the data in buffer.
              if((i > (sizeof(rx_buffer) - 1)))
                  break;
          }else{
              iicSerial2.read();//Put read function here to remove a char "A" in buffer.
          }
      }
      Serial.print("Parsed string: ");
      for(int j = 0; j < i; j++){
          Serial.print(rx_buffer[j]);
      }
      Serial.println();
  }
  delay(1000);
}
```

Expected Result



```
+--
+-----------------------------------------------------------+
|  connected UART2's TX pin to RX pin.                       |
|  Analysis string and eliminate a char of a string.         |
|  Original string: ABCDEFASFGHJUAAAEEB                      |
|  Eliminate char: A                                         |
|  Original string: BCDEFSFGHJUEEB                           |
|  Print the parsed string.                                  |
+-----------------------------------------------------------+
Please Send to the string by UART2's TX.
UART2 send a string: ABCDEFASFGHJUAAAEEB
Parsed string: BCDEFSFGHJUEEB
```

☑ Autoscroll □ Show timestamp Newline 115200 baud

FAQ

1. The module does not work when everything is fine, what should I do?

Check whether all the DIP switches are at the position of "1", when it still does not work after you switched them to 1 already, reset the module by connecting the pinSRT to pin-(negative) for 3s or you can try powering off the module and restart the program.

> For any questions, advice or cool ideas to share, please visit the **DFRobot Forum** (https://www.dfrobot.com/forum)

More Documents

- Schematics Diagram (https://dfimg.dfrobot.com/nobody/wiki/3646cb26e392103672637b458d4995a3.pdf)
- Dimension Diagram (https://dfimg.dfrobot.com/nobody/wiki/53761bd39b49eda1cd3ae0e30c99aba1.pdf)

🛒 Get **Gravity: IIC to Dual UART Module** (https://www.dfrobot.com/product-2001.html) from DFRobot Store or **DFRobot Distributor**. (https://www.dfrobot.com/distributor)

Turn to the Top