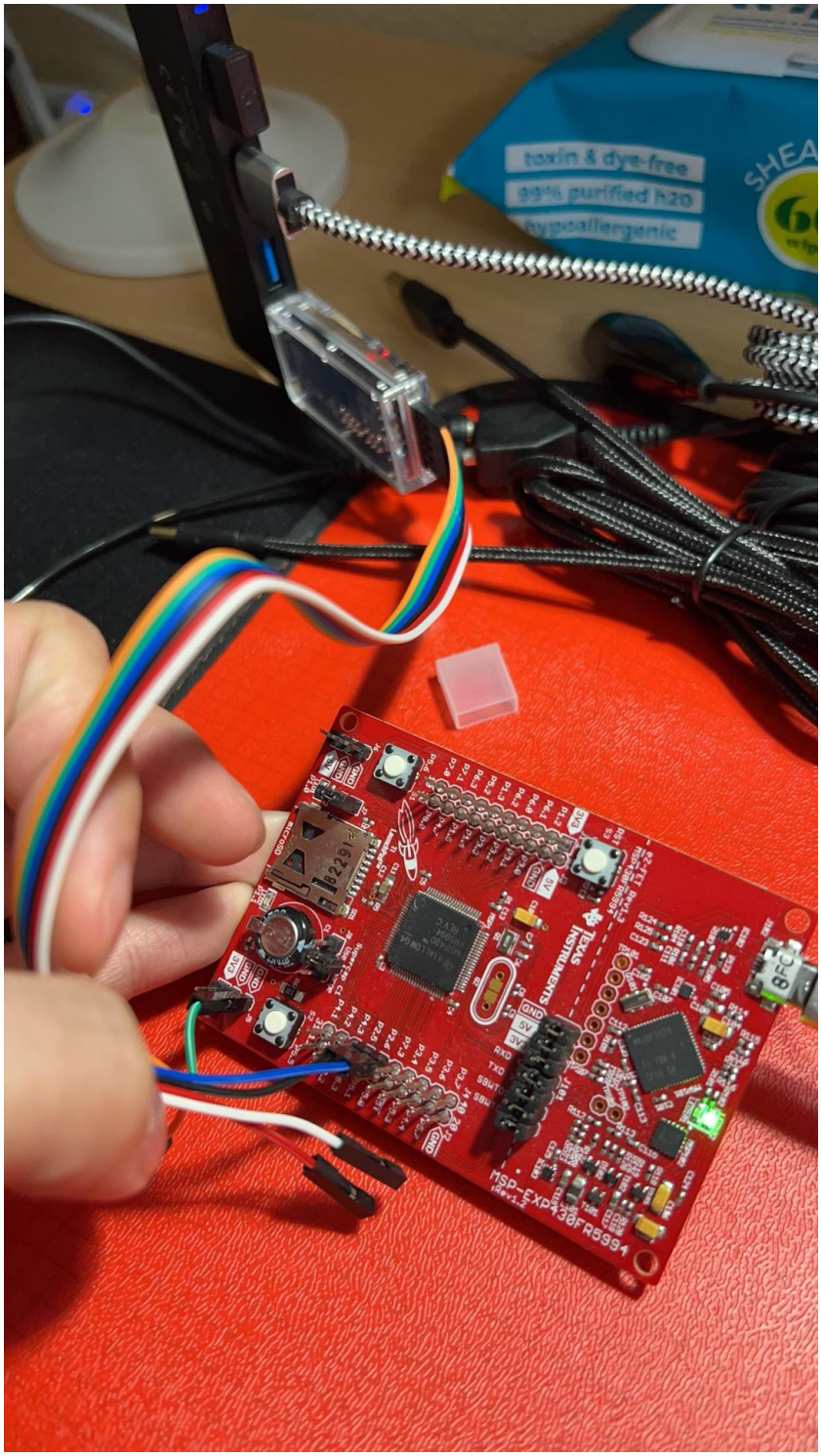
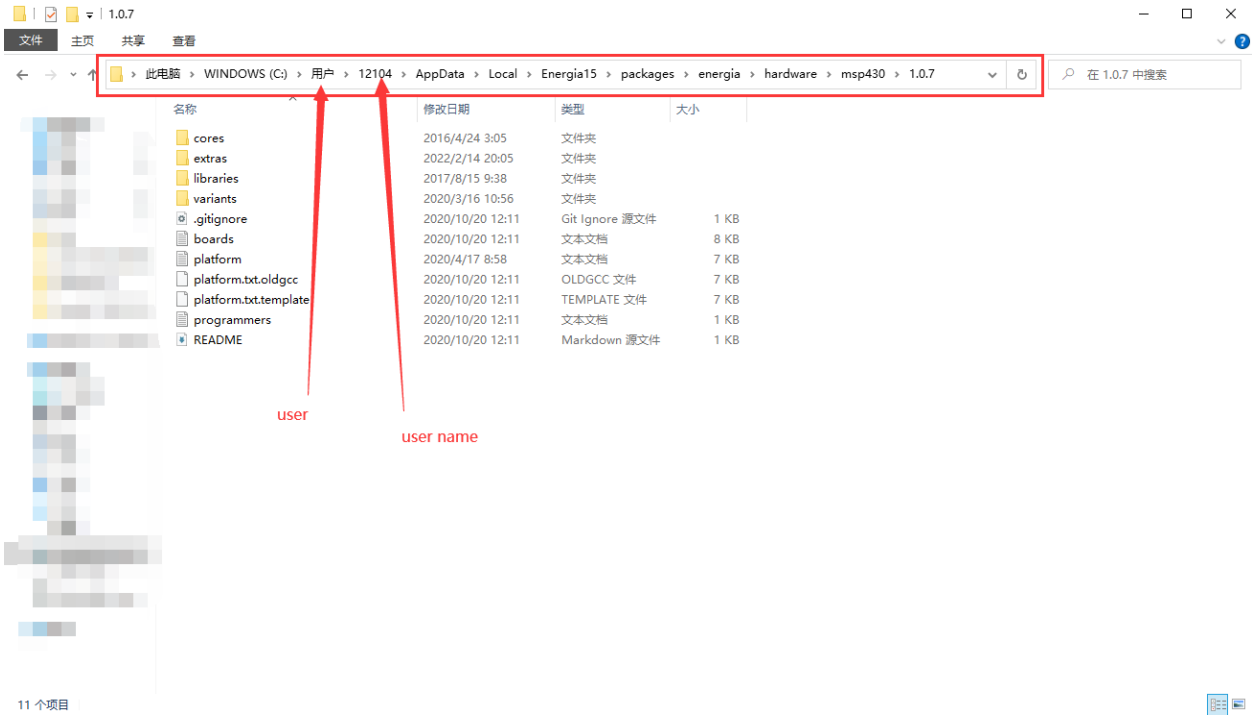
Connect the serial port module and MSP430 to the computer, connect TX to 2.6, RX to 2.5, connect to GND, and do not connect the rest of the wires.



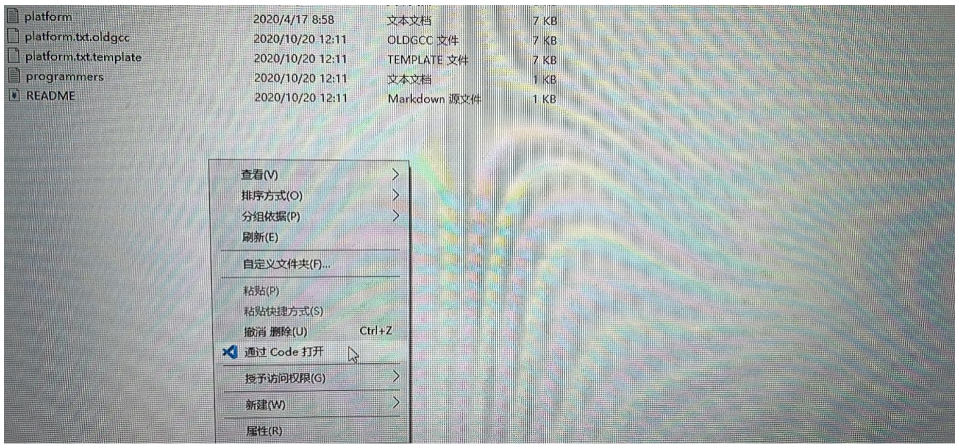
Step 1.

Find file location



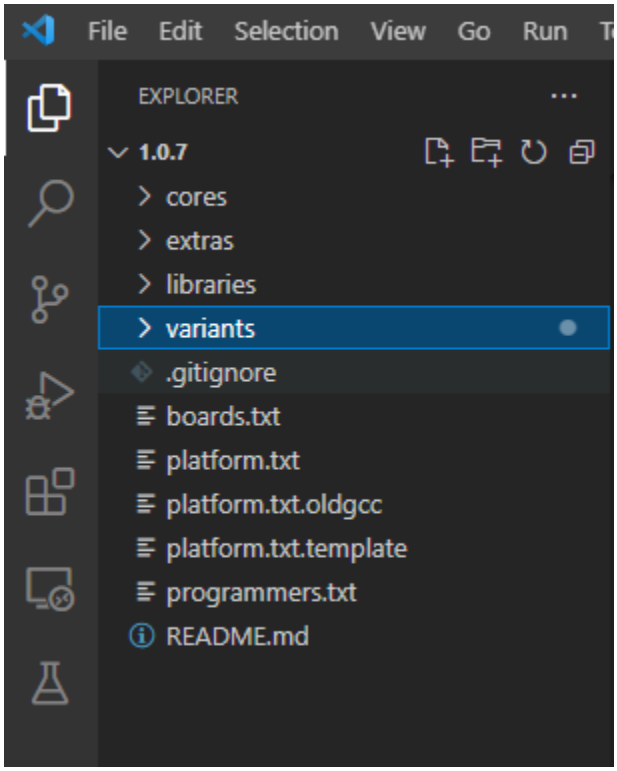
Step 2.

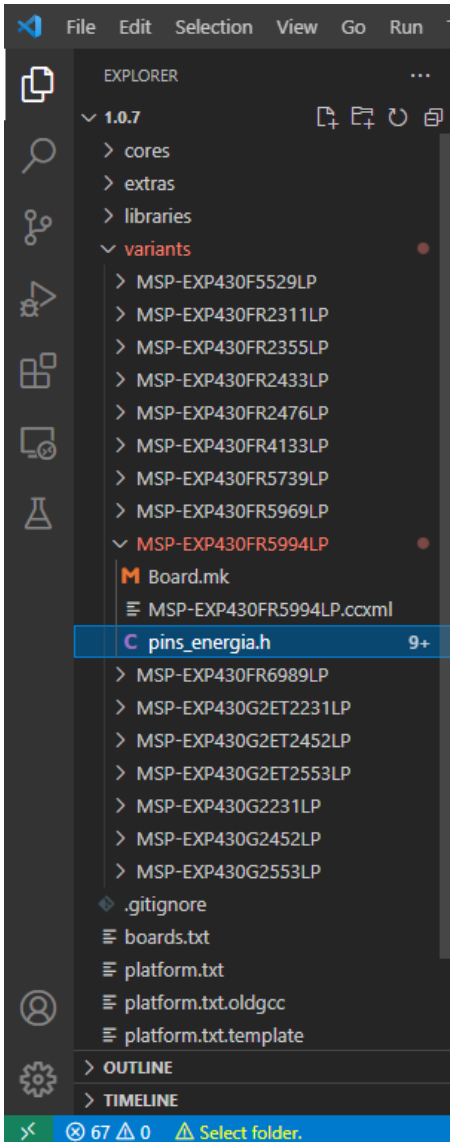
Open via VS code



Step 3.

variants→MSP-EXP430FR5994LP→pins_energia.h→line 97 & 98






```
94  #if defined(__MSP430_HAS_EUSCI_A0__) || defined(__MSP430_HAS_EUSCI_A1__)
95  static const uint8_t DEBUG_UARTRXD = 42;  /* Receive  Data (RXD) at P2.1 */
96  static const uint8_t DEBUG_UARTTXD = 41;  /* Transmit Data (TXD) at P2.0 */
97  static const uint8_t AUX_UARTRXD = 3;     /* Receive  Data (RXD) at P6.1 */
98  static const uint8_t AUX_UARTTXD = 4;     /* Transmit Data (TXD) at P6.0 */
99  #define DEBUG_UARTRXD_SET_MODE (PORT_SELECTION1 | INPUT)
100 #define DEBUG_UARTTXD_SET_MODE (PORT_SELECTION1 | OUTPUT)
101 #define AUX_UARTRXD_SET_MODE (PORT_SELECTION0 | INPUT)
102 #define AUX_UARTTXD_SET_MODE (PORT_SELECTION0 | OUTPUT)
103 #define DEBUG_UART_MODULE_OFFSET 0x00
104 #define AUX_UART_MODULE_OFFSET 0x60
105 #define SERIAL1_AVAILABLE 1
106 #endif
107
108
109 /* Analog pins */
110
```

Step 4.
Following lines 97 and 98, configure P2.5 and P2.6

```
94  #if defined(__MSP430_HAS_EUSCI_A0__) || defined(__MSP430_HAS_EUSCI_A1__)
95  static const uint8_t DEBUG_UARTRXD = 42;  /* Receive  Data (RXD) at P2.1 */
96  static const uint8_t DEBUG_UARTTXD = 41;  /* Transmit Data (TXD) at P2.0 */
97  static const uint8_t AUX_UARTRXD = 3;     /* Receive  Data (RXD) at P6.1 */
98  static const uint8_t AUX_UARTTXD = 4;     /* Transmit Data (TXD) at P6.0 */
99  static const uint8_t AUX_UART2RXD = 34;     /* Receive  Data (RXD) at P2.6 */
100 static const uint8_t AUX_UART2TXD = 35;     /* Transmit Data (TXD) at P2.5 */
```

Step 5.
Following lines 103 and 104

```
101 #define DEBUG_UARTRXD_SET_MODE (PORT_SELECTION1 | INPUT)
102 #define DEBUG_UARTTXD_SET_MODE (PORT_SELECTION1 | OUTPUT)
103 #define AUX_UARTRXD_SET_MODE (PORT_SELECTION0 | INPUT)
104 #define AUX_UARTTXD_SET_MODE (PORT_SELECTION0 | OUTPUT)
```

```
101 #define DEBUG_UARTRXD_SET_MODE (PORT_SELECTION1 | INPUT)
102 #define DEBUG_UARTTXD_SET_MODE (PORT_SELECTION1 | OUTPUT)
103 #define AUX_UARTRXD_SET_MODE (PORT_SELECTION0 | INPUT)
104 #define AUX_UARTTXD_SET_MODE (PORT_SELECTION0 | OUTPUT)
105 #define AUX_UART2RXD_SET_MODE (PORT_SELECTION1 | INPUT)
106 #define AUX_UART2TXD_SET_MODE (PORT_SELECTION1 | OUTPUT)
```

Step 6.

```
107 #define DEBUG_UART_MODULE_OFFSET 0x00
108 #define AUX_UART_MODULE_OFFSET 0x60
```

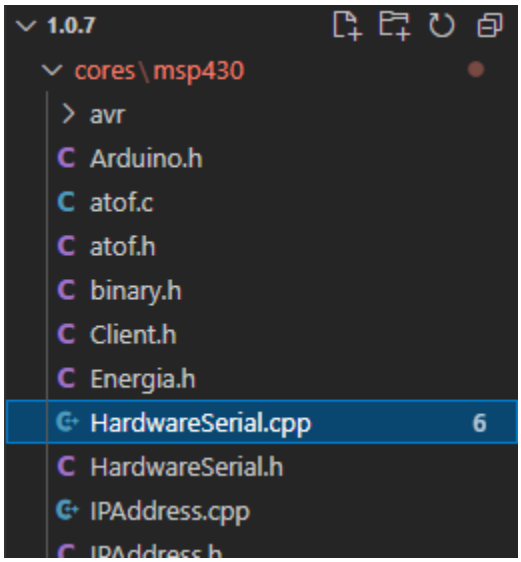
```
107 #define DEBUG_UART_MODULE_OFFSET 0x00
108 #define AUX_UART_MODULE_OFFSET 0x60
109 #define AUX_UART2_MODULE_OFFSET 0x20
```

Then ⇓

```
110 #define SERIAL1_AVAILABLE 1
111 #endif
```

```
110 #define SERIAL1_AVAILABLE 1
111 #define SERIAL2_AVAILABLE 1
112 #endif
```

Step 7.
cores →HardwareSerial.cpp→line 87-90



⇓

```
85  ring_buffer rx_buffer  =  { { 0 }, 0, 0 };
86  ring_buffer tx_buffer  =  { { 0 }, 0, 0 };
87  #ifdef SERIAL1_AVAILABLE
88  ring_buffer rx_buffer1  =  { { 0 }, 0, 0 };
89  ring_buffer tx_buffer1  =  { { 0 }, 0, 0 };
90  #endif
```

```
85  ring_buffer rx_buffer  =  { { 0 }, 0, 0 };
86  ring_buffer tx_buffer  =  { { 0 }, 0, 0 };
87  #ifdef SERIAL1_AVAILABLE
88  ring_buffer rx_buffer1  =  { { 0 }, 0, 0 };
89  ring_buffer tx_buffer1  =  { { 0 }, 0, 0 };
90  #endif
91  #ifdef SERIAL2_AVAILABLE
92  ring_buffer rx_buffer2  =  { { 0 }, 0, 0 };
93  ring_buffer tx_buffer2  =  { { 0 }, 0, 0 };
94  #endif
95
```

Step 8.
Lines 112-115

```
110  void serialEvent() __attribute__((weak));
111  void serialEvent() {}
112  #ifdef SERIAL1_AVAILABLE
113  void serialEvent1() __attribute__((weak));
114  void serialEvent1() {}
115  #endif
116
```

```
110  void serialEvent() __attribute__((weak));
111  void serialEvent() {}
112  #ifdef SERIAL1_AVAILABLE
113  void serialEvent1() __attribute__((weak));
114  void serialEvent1() {}
115  #endif
116  #ifdef SERIAL2_AVAILABLE
117  void serialEvent2() __attribute__((weak));
118  void serialEvent2() {}
119  #endif
```

Step 9.
Lines 124-125

```
121  void serialEventRun(void)
122  {
123    if (Serial.available()) serialEvent();
124  #ifdef SERIAL1_AVAILABLE
125    if (Serial1.available()) serialEvent1();
126  #endif
127  }
```

```
121  void serialEventRun(void)
122  {
123      if (Serial.available()) serialEvent();
124  #ifdef SERIAL1_AVAILABLE
125      if (Serial1.available()) serialEvent1();
126  #endif
127  #ifdef SERIAL2_AVAILABLE
128      if (Serial2.available()) serialEvent2();
129  #endif
130  }
```

Step 10.
Comment out lines 265-270 and modify them (here determines the attribution of information. The reason why our previous operation failed is because although all three serial ports are enabled, there is no correct judgment on the attribution of information. The commented-out paragraph is to judge whether the information is Serial, if not, all are counted as Serial1, resulting in no information in Sreial2.)

```
274 /*#ifdef SERIAL1_AVAILABLE
275     // Debug uart aka Serial always gets rx_buffer and aux aka Serial1 gets rx_buffer1
276     ring_buffer *rx_buffer_ptr = (offset == DEBUG_UART_MODULE_OFFSET) ? &rx_buffer:&rx_buffer1;
277 #else
278     ring_buffer *rx_buffer_ptr = &rx_buffer;
279 #endif*/
```

⇓

Add judgment (rx) statement

```
258 HardwareSerial::operator bool() {
259     return true;
260 }
261
262 void uart_rx_isr(uint16_t offset)
263 {
264     ring_buffer *rx_buffer_ptr;
265     if(offset == AUX_UART_MODULE_OFFSET){
266         rx_buffer_ptr = &rx_buffer1;
267     }
268     else if(offset == AUX_UART2_MODULE_OFFSET){
269         rx_buffer_ptr = &rx_buffer2;
270     }
271     else{
272         rx_buffer_ptr = &rx_buffer;
273     }
274 /*#ifdef SERIAL1_AVAILABLE
275     // Debug uart aka Serial always gets rx_buffer and aux aka Serial1 gets rx_buffer1
276     ring_buffer *rx_buffer_ptr = (offset == DEBUG_UART_MODULE_OFFSET) ? &rx_buffer:&rx_buffer1;
277 #else
278     ring_buffer *rx_buffer_ptr = &rx_buffer;
279 #endif*/
```

Step 11.
Comment out lines 286-291

```
284 void uart_tx_isr(uint16_t offset)
285 {
286 /*#ifdef SERIAL1_AVAILABLE
287     // Debug uart aka Serial always gets rx_buffer and aux aka Serial1 gets rx_buffer1
288     ring_buffer *tx_buffer_ptr = (offset == DEBUG_UART_MODULE_OFFSET) ? &tx_buffer : &tx_buffer1;
289 #else
290     ring_buffer *tx_buffer_ptr = &tx_buffer;
291 #endif*/
```

Step 12.
Copy lines 264-273

```
263   {
264       ring_buffer *rx_buffer_ptr;
265       if(offset == AUX_UART_MODULE_OFFSET){
266           rx_buffer_ptr = &rx_buffer1;
267       }
268       else if(offset == AUX_UART2_MODULE_OFFSET){
269           rx_buffer_ptr = &rx_buffer2;
270       }
271       else{
272           rx_buffer_ptr = &rx_buffer;
273       }
274   /*#ifdef SERIAL1_AVAILABLE
```

⇓ Paste it to the next line of "{" on line 285, and change rx to tx

```
284   void uart_tx_isr(uint16_t offset)
285   {
286       ring_buffer *tx_buffer_ptr;
287       if(offset == AUX_UART_MODULE_OFFSET){
288           tx_buffer_ptr = &tx_buffer1;
289       }
290       else if(offset == AUX_UART2_MODULE_OFFSET){
291           tx_buffer_ptr = &tx_buffer2;
292       }
293       else{
294           tx_buffer_ptr = &tx_buffer;
295       }
296   /*#ifdef SERIAL1_AVAILABLE
297       // Debug uart aka Serial always gets rx_buffer and aux aka Serial1 gets rx_buffer1
298       ring_buffer *tx_buffer_ptr = (offset == DEBUG_UART_MODULE_OFFSET) ? &tx_buffer : &tx_buffer1;
299   #else
300       ring_buffer *tx_buffer_ptr = &tx_buffer;
301   #endif*/
```

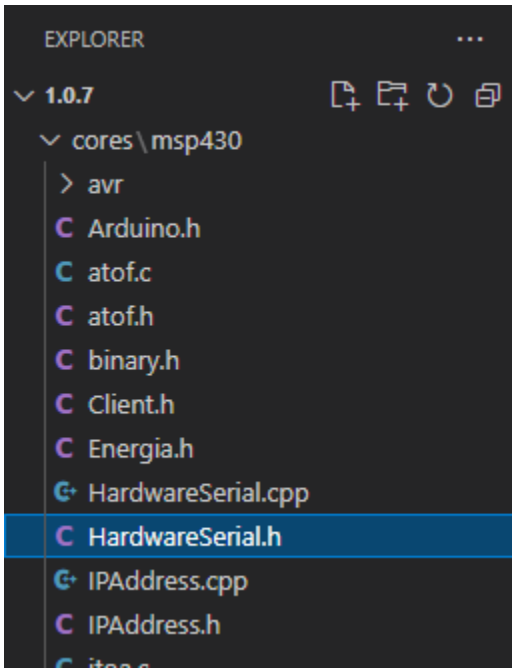
Step 13.
pull to the end

```
299   rdwareSerial Serial(&rx_buffer, &tx_buffer, DEBUG_UART_MODULE_OFFSET, DEBUG_UARTRXD_SET_MODE, DEBUG_UARTTXD_SET_MODE, DEBUG_UARTRXD, DEBUG_UARTTXD);
300   #ifdef SERIAL1_AVAILABLE
301   HardwareSerial Serial1(&rx_buffer1, &tx_buffer1, AUX_UART_MODULE_OFFSET, AUX_UARTRXD_SET_MODE, AUX_UARTTXD_SET_MODE, AUX_UARTRXD, AUX_UARTTXD);
302   #endif
303
```

⇓

```
319   HardwareSerial Serial(&rx_buffer, &tx_buffer, DEBUG_UART_MODULE_OFFSET, DEBUG_UARTRXD_SET_MODE, DEBUG_UARTTXD_SET_MODE, DEBUG_UARTRXD, DEBUG_UARTTXD);
320   #ifdef SERIAL1_AVAILABLE
321   HardwareSerial Serial1(&rx_buffer1, &tx_buffer1, AUX_UART_MODULE_OFFSET, AUX_UARTRXD_SET_MODE, AUX_UARTTXD_SET_MODE, AUX_UARTRXD, AUX_UARTTXD);
322   #endif
323   #ifdef SERIAL2_AVAILABLE
324   HardwareSerial Serial2(&rx_buffer2, &tx_buffer2, AUX_UART2_MODULE_OFFSET, AUX_UART2RXD_SET_MODE, AUX_UART2TXD_SET_MODE, AUX_UART2RXD, AUX_UART2TXD);
325   #endif
326   #endif
327
```

Step 14.
HardwareSerial.h→line 100



⇓