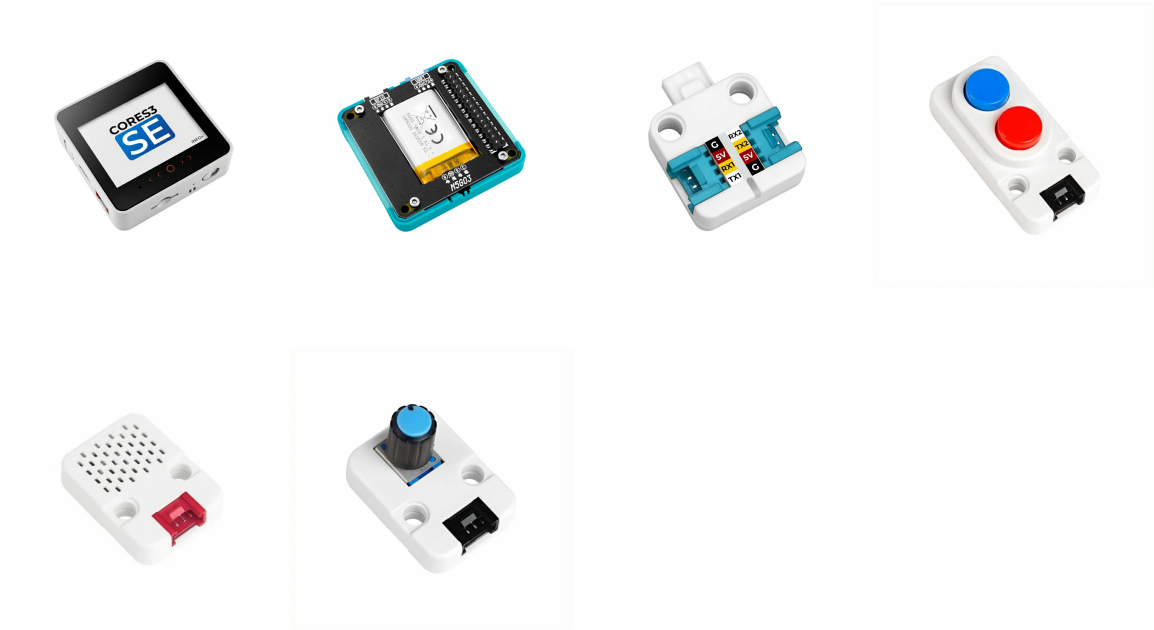


Unit ChainBus Arduino 使用教程

1. 准备工作

- 环境配置：参考 [Arduino IDE 上手教程](#) 完成 IDE 安装，并根据实际使用的开发板安装对应的板管理，与需要的驱动库。
- 使用到的驱动库：
 - [M5Unified](#)
 - [M5Chain](#)
- 使用到的硬件产品：
 - [CoreS3-SE](#)
 - [Base M5GO Bottom3](#)
 - [Unit ChainBus](#)
 - [Unit Dual Button](#)
 - [Unit ENV-III](#)
 - [Unit Angle](#)



2. 注意事项

引脚兼容性

由于每款主机的引脚配置不同，使用前请参考产品文档中的[引脚兼容表](#)，并根据实际引脚连接情况修改案例程序。

3. 编译上传

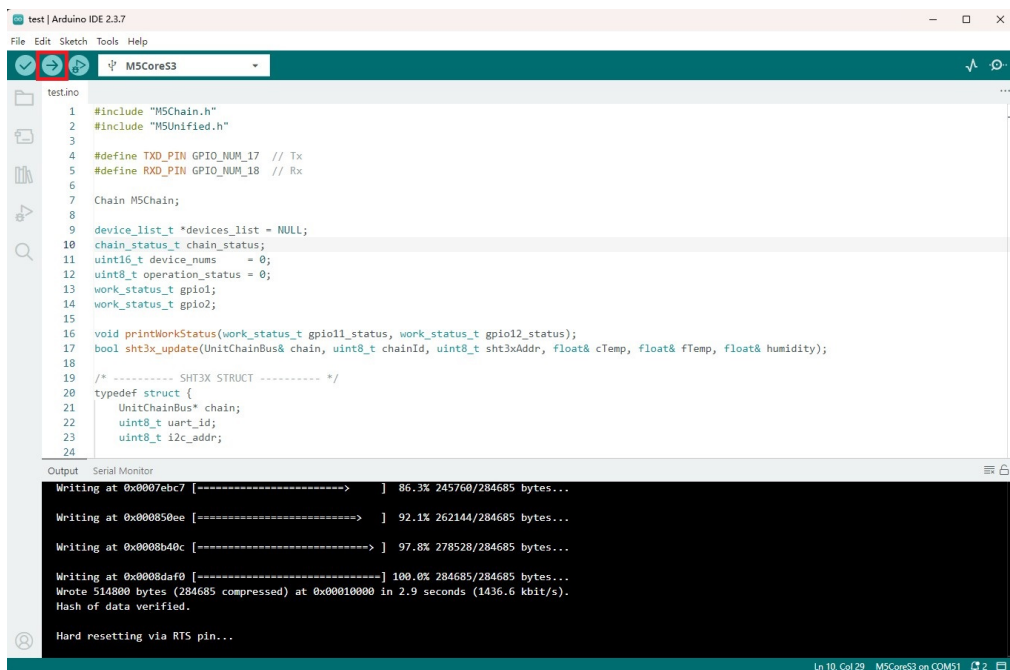
- 1. 进入下载模式：CoreS3-SE 长按复位按键 (大约 2 秒) 直到内部绿色 LED 灯亮起，便可松开，此时设备已进入下载模式，等待烧录。

说明

不同设备进行程序烧录前需要进入下载模式，不同的主控设备该步骤可能有所不同。详情可参考[Arduino IDE上手教程](#)页面底部的设备程序下载教程列表，查看具体的操作方式。



- 2. 选中设备端口，点击 Arduino IDE 左上角编译上传按钮，等待程序完成编译并上传至设备。



4. 案例程序

- 本教程中使用的主控设备为 CoreS3-SE，用 Base M5GO Bottom3 底座供电，使用 PORT.C，搭配 Unit CHainBus 及其他模块实现效果。本模块采用串口的方式通讯，根据实际的电路连接修改程序中的引脚定义，设备连接后对应的串口 IO 为 G17（RX），G18（TX）。

说明

主机未断电重启时，Unit ChainBus 的 GPIO 的工作模式会保持上次设置的状态，例如 GPIO2 设置为输入模式，主机未断电重启后，即使没有重新设置 GPIO2 的工作模式，GPIO2 仍然会保持输入模式。

4.1 I2C 模式

```
1  #include "M5Chain.h"
2  #include "M5Unified.h"
3
4  #define TXD_PIN GPIO_NUM_17 // Tx
5  #define RXD_PIN GPIO_NUM_18 // Rx
6
7  Chain M5Chain;
8
9  device_list_t *devices_list = NULL;
10 chain_status_t chain_status;
11 uint16_t device_nums = 0;
12 uint8_t operation_status = 0;
13 work_status_t gpio1;
14 work_status_t gpio2;
15
16 void printWorkStatus(String gpio_num, work_status_t gpio_status);
17 bool sht3x_update(UnitChainBus& chain, uint8_t chainId, uint8_t sht3xAddr, float& cTemp, float& fTemp, float& humidity);
18
19 /* ----- SHT3X STRUCT ----- */
20 typedef struct {
21     UnitChainBus* chain;
22     uint8_t uart_id;
23     uint8_t i2c_addr;
24
25     float cTemp;
26     float fTemp;
27     float humidity;
28
29     bool update()
30     {
31         if (chain == nullptr) return false;
32         return sht3x_update(*chain, uart_id, i2c_addr,
33                             cTemp, fTemp, humidity);
34     }
35 } SHT3X_t;
36
37 SHT3X_t sht3x;
38
39 void setup()
40 {
41     M5.begin();
42     M5.Display.clear();
43     M5.Display.setFont(&FreeMonoBold12pt7b);
44     M5.Display.setTextDatum(middle_center);
45     M5.Display.drawCenterString("Unit ChainBus Test", 160, 120);
46
47     Serial.begin(115200);
48     Serial.println("Unit ChainBus Test");
49     M5Chain.begin(&Serial2, 115200, RXD_PIN, TXD_PIN);
50 }
```

```

51 // Check whether any Unit Chain device is connected
52 if (M5Chain.isDeviceConnected()) {
53     Serial.println("devices is connected");
54     chain_status = M5Chain.getDeviceNum(&device_nums);
55     if (chain_status == CHAIN_OK) {
56         devices_list = (device_list_t *)malloc(sizeof(device_list_t));
57         devices_list->count = device_nums;
58         devices_list->devices = (device_info_t *)malloc(sizeof(device_info_t) * device_nums);
59         // Get detailed device list
60         if (M5Chain.getDeviceList(devices_list)) {
61             Serial.println("devices list get success");
62         } else {
63             Serial.println("devices list get failed");
64         }
65     } else {
66         Serial.printf("error status:%d \r\n", chain_status);
67         Serial.printf("devices num get failed.\r\n");
68     }
69 } else {
70     Serial.println("devices is not connected.");
71 }
72
73 if (devices_list->devices[0].device_type == UNIT_CHAIN_BUS_TYPE_CODE) {
74     Serial.println("ID[1] is Unit ChainBus\n");
75     chain_status = M5Chain.setChainBusI2cMode(devices_list->devices[0].id, CHAIN_I2C_HIGH_SPEED);
76     if (chain_status == CHAIN_OK && operation_status == 1) {
77         Serial.printf("Unit ChainBus ID[%d] set i2c mode success \r\n", devices_list->devices[0].id);
78         M5Chain.getChainBusWorkMode(devices_list->devices[0].id, &gpio1, &gpio2); // Read Unit ChainBus WorkMode
79         printWorkStatus("GPIO_1", (work_status_t)gpio1);
80         printWorkStatus("GPIO_2", (work_status_t)gpio2);
81     } else {
82         Serial.printf("Unit ChainBus ID[%d] set i2c mode fail, chain_status:%d operation_status:%d\n", devices_list->devices[0].id, chain_status, operation_status);
83     }
84     delay(500);
85
86     // Bind SHT3X to this ChainBus
87     sht3x.chain = &M5Chain;
88     sht3x.uart_id = devices_list->devices[0].id;
89     sht3x.i2c_addr = 0x44; // SHT3X I2C address, can get from getChainBusI2cScanAddr
90
91     Serial.printf("\nUnit ENV-III SHT3X bind to ChainBus ID[%d]\r\n", sht3x.uart_id);
92 } else {
93     Serial.println("ID[1] is NOT Unit ChainBus\n");
94     return;
95 }
96
97 delay(5);
98 }
99
100 void loop()
101 {
102     if (sht3x.update()) {
103         Serial.println("-----SHT3X-----");

```



```

104     Serial.print("Temperature: ");
105     Serial.print(sht3x.cTemp);
106     Serial.println(" 'C");
107     Serial.print("Humidity: ");
108     Serial.print(sht3x.humidity);
109     Serial.println("% rH");
110     Serial.println("-----\r\n");
111
112     M5.Display.fillRect(0, 0, 320, 240, TFT_BLACK);
113     M5.Display.setCursor(0, 20);
114     M5.Display.println("-----SHT3X-----");
115     M5.Display.print("Temperature: ");
116     M5.Display.print(sht3x.cTemp);
117     M5.Display.println(" 'C");
118     M5.Display.print("Humidity: ");
119     M5.Display.print(sht3x.humidity);
120     M5.Display.println("% rH");
121     M5.Display.println("-----\r\n");
122 }
123
124 delay(1000);
125 }
126
127 void printWorkStatus(String gpio_num, work_status_t gpio_status)
128 {
129     Serial.print(" >>> " + gpio_num + ": ");
130     switch (gpio_status) {
131         case CHAIN_NOT_WORK_STATUS:
132             Serial.println("Not configured working status");
133             break;
134         case CHAIN_OUTPUT_WORK_STATUS:
135             Serial.println("Output status");
136             break;
137         case CHAIN_INPUT_WORK_STATUS:
138             Serial.println("Input status");
139             break;
140         case CHAIN_NVIC_WORK_STATUS:
141             Serial.println("External interrupt working status");
142             break;
143         case CHAIN_ADC_WORK_STATUS:
144             Serial.println("ADC working status");
145             break;
146         case CHAIN_I2C_WORK_STATUS:
147             Serial.println("I2C working status");
148             break;
149         default:
150             Serial.println("Unrecognized work status");
151             break;
152     }
153 }
154
155 // Perform one complete SHT3X measurement cycle via Unit ChainBus I2C
156 bool sht3x_update(UnitChainBus& chain,

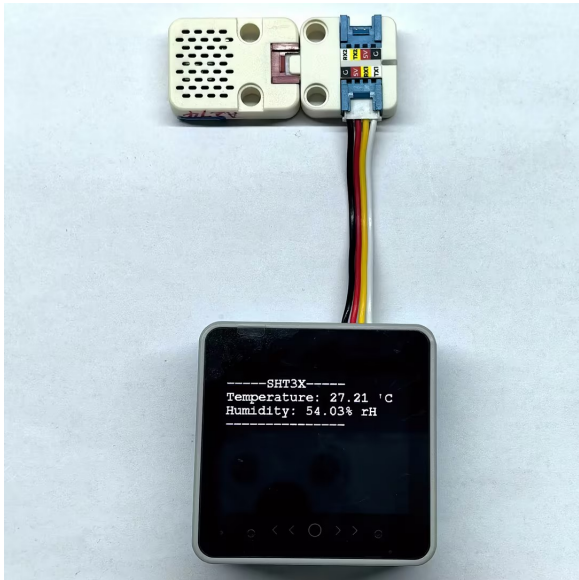
```

```

157         uint8_t chainId,
158         uint8_t sht3xAddr,
159         float& cTemp,
160         float& fTemp,
161         float& humidity)
162 {
163     uint8_t opStatus = 0;
164     chain_status_t status;
165
166     // 1. Send measurement command (high repeatability, clock stretching disabled)
167     uint8_t cmd[2] = {0x2C, 0x06};
168     status = chain.chainBusI2cWrite(
169         chainId,
170         sht3xAddr,
171         2,
172         cmd,
173         &opStatus,
174         1000
175     );
176
177     if (status != CHAIN_OK || opStatus != CHAIN_BUS_OPERATION_SUCCESS) {
178         return false;
179     }
180
181     // 2. Wait for the measurement to complete
182     delay(100);
183
184     // 3. Read 6 bytes of measurement data
185     uint8_t data[6] = {0};
186     status = chain.chainBusI2cRead(
187         chainId,
188         sht3xAddr,
189         6,
190         data,
191         &opStatus,
192         1000
193     );
194
195     if (status != CHAIN_OK || opStatus != CHAIN_BUS_OPERATION_SUCCESS) {
196         return false;
197     }
198
199     // 4. Convert raw data to temperature and humidity
200     cTemp = (((data[0] * 256.0) + data[1]) * 175.0) / 65535.0 - 45.0;
201     fTemp = (cTemp * 1.8) + 32.0;
202     humidity = (((data[3] * 256.0) + data[4]) * 100.0) / 65535.0;
203
204     return true;
205 }

```

设备上电后，串口监视器会输出连接设备的信息，并通过 Unit ChainBus 的接口读取连接在其上的 Unit ENV-III 的 SHT3X 传感器温湿度数据。



串口监视器输出示例：

```
Unit ChainBus Test
devices is connected
devices list get success
ID[1] is Unit ChainBus

Unit ChainBus ID[1] set i2c mode success
>>> GPIO_1: I2C working status
>>> GPIO_2: I2C working status

Unit ENV-III SHT3X bind to ChainBus ID[1]
-----SHT3X-----
Temperature: 27.21 'C
Humidity: 54.03% rH
-----
```

4.2 输入模式

```
1  #include "M5Chain.h"
2  #include "M5Unified.h"
3
4  #define TXD_PIN GPIO_NUM_17 // Tx
5  #define RXD_PIN GPIO_NUM_18 // Rx
6
7  Chain M5Chain;
8
9  device_list_t *devices_list = NULL;
10 chain_status_t chain_status;
11 uint16_t device_nums = 0;
12 uint8_t operation_status = 0;
13 work_status_t gpio1;
14 work_status_t gpio2;
15
16 void printWorkStatus(String gpio_num, work_status_t gpio_status);
17
18 void setup()
19 {
20     M5.begin();
21     M5.Display.clear();
22     M5.Display.setFont(&FreeMonoBold12pt7b);
23     M5.Display.setTextDatum(middle_center);
24     M5.Display.drawCenterString("Unit ChainBus Test", 160, 120);
25
26     Serial.begin(115200);
27     Serial.println("Unit ChainBus Test");
28     M5Chain.begin(&Serial2, 115200, RXD_PIN, TXD_PIN);
29
30     // Check whether any Unit Chain device is connected
31     if (M5Chain.isDeviceConnected()) {
32         Serial.println("devices is connected");
33         chain_status = M5Chain.getDeviceNum(&device_nums);
34         if (chain_status == CHAIN_OK) {
35             devices_list = (device_list_t *)malloc(sizeof(device_list_t));
36             devices_list->count = device_nums;
37             devices_list->devices = (device_info_t *)malloc(sizeof(device_info_t) * device_nums);
38             // Get detailed device list
39             if (M5Chain.getDeviceList(devices_list)) {
40                 Serial.println("devices list get success");
41             } else {
42                 Serial.println("devices list get failed");
43             }
44         } else {
45             Serial.printf("error status:%d \r\n", chain_status);
46             Serial.printf("devices num get failed.\r\n");
47         }
48     } else {
49         Serial.println("devices is not connected.");
50     }
```

```

51
52     if (devices_list->devices[0].device_type == UNIT_CHAIN_BUS_TYPE_CODE) {
53         Serial.println("ID[1] is Unit ChainBus\n");
54         chain_status = M5Chain.setChainBusInputMode(devices_list->devices[0].id, CHAIN_GPIO_PIN_1, (
55         if (chain_status == CHAIN_OK && operation_status == 1) {
56             Serial.printf("Unit ChainBus ID[%d] set GPIO1 input mode success \r\n", devices_list->de
57             M5Chain.getChainBusWorkMode(devices_list->devices[0].id, &gpio1, &gpio2); // Read Unit C
58             printWorkStatus("GPIO_1", (work_status_t)gpio1);
59             printWorkStatus("GPIO_2", (work_status_t)gpio2);
60         } else {
61             Serial.printf("Unit ChainBus ID[%d] set GPIO1 input mode fail, chain_status:%d  operati
62         }
63         delay(500);
64     } else {
65         Serial.println("ID[1] is NOT Unit ChainBus\n");
66         return;
67     }
68
69     delay(5);
70 }
71
72 uint8_t last_button_status = true;
73
74 void loop()
75 {
76     uint8_t button_status;
77     // Read GPIO1 input level from Unit ChainBus
78     chain_status = M5Chain.getChainBusInputLevel(devices_list->devices[0].id, CHAIN_GPIO_PIN_1, &but
79     if (chain_status == CHAIN_OK && operation_status == 1) {
80         if (button_status != last_button_status) {
81             Serial.printf("button status: %d \r\n", button_status);
82             last_button_status = button_status;
83         }
84     } else {
85         Serial.printf("get button status failed, chain_status:%d  operation_status:%d \r\n", chain_
86             operation_status);
87     }
88
89     delay(20);
90 }
91
92 void printWorkStatus(String gpio_num, work_status_t gpio_status)
93 {
94     Serial.print(" >>> " + gpio_num + ": ");
95     switch (gpio_status) {
96         case CHAIN_NOT_WORK_STATUS:
97             Serial.println("Not configured working status");
98             break;
99         case CHAIN_OUTPUT_WORK_STATUS:
100             Serial.println("Output status");
101             break;
102         case CHAIN_INPUT_WORK_STATUS:
103             Serial.println("Input status");

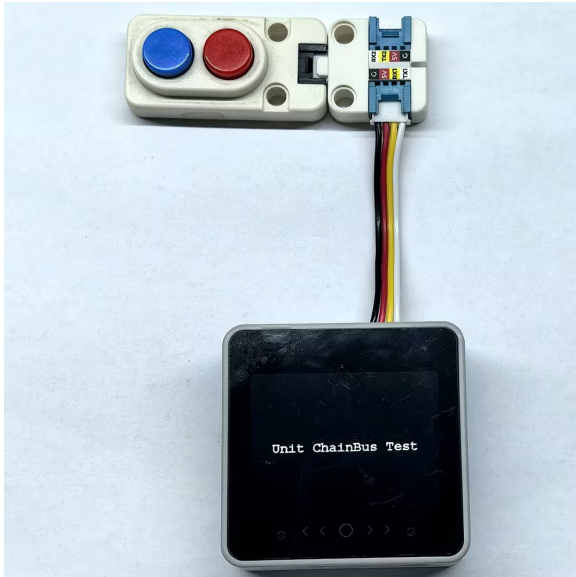
```

```

104         break;
105     case CHAIN_NVIC_WORK_STATUS:
106         Serial.println("External interrupt working status");
107         break;
108     case CHAIN_ADC_WORK_STATUS:
109         Serial.println("ADC working status");
110         break;
111     case CHAIN_I2C_WORK_STATUS:
112         Serial.println("I2C working status");
113         break;
114     default:
115         Serial.println("Unrecognized work status");
116         break;
117 }
118 }

```

设备上电后，串口监视器会输出连接设备的信息，并通过 Unit ChainBus 的接口读取连接在其上的 Unit Dual Button 蓝色按钮状态变化。



串口监视器输出示例：

```

Unit ChainBus Test
devices is connected
devices list get success
ID[1] is Unit ChainBus

Unit ChainBus ID[1] set GPIO1 input mode success
>>> GPIO_1: Input status
>>> GPIO_2: Not configured working status
button status: 0
button status: 1

```

4.3 输出模式

下方程序中同时控制了 Unit ChainBus 的 GPIO1 和 GPIO2，其中 GPIO1 每 500ms 切换一次高低电平，GPIO2 设置为低电平，可连接 LED 灯进行观察。

```
1  #include "M5Chain.h"
2  #include "M5Unified.h"
3
4  #define TXD_PIN GPIO_NUM_17 // Tx
5  #define RXD_PIN GPIO_NUM_18 // Rx
6
7  Chain M5Chain;
8
9  device_list_t *devices_list = NULL;
10 chain_status_t chain_status;
11 uint16_t device_nums = 0;
12 uint8_t operation_status = 0;
13 work_status_t gpio1;
14 work_status_t gpio2;
15
16 void printWorkStatus(String gpio_num, work_status_t gpio_status);
17
18 void setup()
19 {
20     M5.begin();
21     M5.Display.clear();
22     M5.Display.setFont(&FreeMonoBold12pt7b);
23     M5.Display.setTextDatum(middle_center);
24     M5.Display.drawCenterString("Unit ChainBus Test", 160, 120);
25
26     Serial.begin(115200);
27     Serial.println("Unit ChainBus Test");
28     M5Chain.begin(&Serial2, 115200, RXD_PIN, TXD_PIN);
29
30     if (M5Chain.isDeviceConnected()) {
31         Serial.println("devices is connected");
32         chain_status = M5Chain.getDeviceNum(&device_nums);
33         if (chain_status == CHAIN_OK) {
34             devices_list = (device_list_t *)malloc(sizeof(device_list_t));
35             devices_list->count = device_nums;
36             devices_list->devices = (device_info_t *)malloc(sizeof(device_info_t) * device_nums);
37             if (M5Chain.getDeviceList(devices_list)) {
38                 Serial.println("devices list get success");
39             } else {
40                 Serial.println("devices list get failed");
41             }
42         } else {
43             Serial.printf("error status:%d \r\n", chain_status);
44             Serial.printf("devices num get failed.\r\n");
45         }
46     } else {
47         Serial.println("devices is not connected.");
48     }
49
50     if (devices_list->devices[0].device_type == UNIT_CHAIN_BUS_TYPE_CODE) {
```

```

51     Serial.println("ID[1] is Unit ChainBus\n");
52     chain_status = M5Chain.setChainBusOutputMode(devices_list->devices[0].id, CHAIN_GPIO_PIN_1,
53     chain_status = (chain_status_t)(chain_status &
54         M5Chain.setChainBusOutputMode(devices_list->devices[0].id, CHAIN_GPIO_PIN_2, CHA
55     if (chain_status == CHAIN_OK && operation_status == 1) {
56         Serial.printf("Unit ChainBus ID[%d] set output mode success \r\n", devices_list->device:
57         M5Chain.getChainBusWorkMode(devices_list->devices[0].id, &gpio1, &gpio2); // Read Unit C
58         printWorkStatus("GPIO_1", (work_status_t)gpio1);
59         printWorkStatus("GPIO_2", (work_status_t)gpio2);
60         M5Chain.setChainBusOutputLevel(devices_list->devices[0].id, CHAIN_GPIO_PIN_2, CHAIN_GPIO
61     } else {
62         Serial.printf("Unit ChainBus ID[%d] set output mode fail, chain_status:%d  operation_st:
63     }
64     delay(500);
65 } else {
66     Serial.println("ID[1] is NOT Unit ChainBus\n");
67     return;
68 }
69
70 delay(5);
71 }
72
73 bool flag = false;
74
75 void loop()
76 {
77     // change GPIO1 output level
78     flag = !flag;
79     chain_status = M5Chain.setChainBusOutputLevel(devices_list->devices[0].id, CHAIN_GPIO_PIN_1, fla
80     if (chain_status == CHAIN_OK && operation_status == 1) {
81         if (flag) {
82             Serial.println("set gpio1 high level");
83         } else {
84             Serial.println("set gpio1 low level");
85         }
86     } else {
87         Serial.printf("set gpio1 level failed, chain_status:%d  operation_status:%d \r\n", chain_st:
88     }
89
90     delay(500);
91 }
92
93 void printWorkStatus(String gpio_num, work_status_t gpio_status)
94 {
95     Serial.print(" >>> " + gpio_num + ": ");
96     switch (gpio_status) {
97         case CHAIN_NOT_WORK_STATUS:
98             Serial.println("Not configured working status");
99             break;
100        case CHAIN_OUTPUT_WORK_STATUS:
101            Serial.println("Output status");
102            break;
103        case CHAIN_INPUT_WORK_STATUS:

```



```

104         Serial.println("Input status");
105         break;
106     case CHAIN_NVIC_WORK_STATUS:
107         Serial.println("External interrupt working status");
108         break;
109     case CHAIN_ADC_WORK_STATUS:
110         Serial.println("ADC working status");
111         break;
112     case CHAIN_I2C_WORK_STATUS:
113         Serial.println("I2C working status");
114         break;
115     default:
116         Serial.println("Unrecognized work status");
117         break;
118     }
119 }

```

设备上电后，串口监视器会输出连接设备的信息，并通过 Unit ChainBus 的接口控制连接在其上的 LED 灯的闪烁。

[Unit_ChainBus_Arduino_output.mp4](#)

串口监视器输出示例：

```

Unit ChainBus Test
devices is connected
devices list get success
ID[1] is Unit ChainBus

Unit ChainBus ID[1] set output mode success
>>> GPIO_1: Output status
>>> GPIO_2: Output status
set gpio1 high level
set gpio1 low level
set gpio1 high level
set gpio1 low level

```

4.4 中断模式

```
1  #include "M5Chain.h"
2  #include "M5Unified.h"
3
4  #define TXD_PIN GPIO_NUM_17 // Tx
5  #define RXD_PIN GPIO_NUM_18 // Rx
6
7  Chain M5Chain;
8
9  device_list_t *devices_list = NULL;
10 chain_status_t chain_status;
11 uint16_t device_nums = 0;
12 uint8_t operation_status = 0;
13 work_status_t gpio1;
14 work_status_t gpio2;
15
16 void printWorkStatus(String gpio_num, work_status_t gpio_status);
17
18 void setup()
19 {
20     M5.begin();
21     M5.Display.clear();
22     M5.Display.setFont(&FreeMonoBold12pt7b);
23     M5.Display.setTextDatum(middle_center);
24     M5.Display.drawCenterString("Unit ChainBus Nvic Test", 160, 120);
25
26     Serial.begin(115200);
27     Serial.println("Unit ChainBus Nvic Test");
28     M5Chain.begin(&Serial2, 115200, RXD_PIN, TXD_PIN);
29
30     if (M5Chain.isDeviceConnected()) {
31         Serial.println("devices is connected");
32         chain_status = M5Chain.getDeviceNum(&device_nums);
33         if (chain_status == CHAIN_OK) {
34             devices_list = (device_list_t *)malloc(sizeof(device_list_t));
35             devices_list->count = device_nums;
36             devices_list->devices = (device_info_t *)malloc(sizeof(device_info_t) * device_nums);
37             if (M5Chain.getDeviceList(devices_list)) {
38                 Serial.println("devices list get success");
39             } else {
40                 Serial.println("devices list get failed");
41             }
42         } else {
43             Serial.printf("error status:%d \r\n", chain_status);
44             Serial.printf("devices num get failed.\r\n");
45         }
46     } else {
47         Serial.println("devices is not connected.");
48     }
49
50     if (devices_list->devices[0].device_type == UNIT_CHAIN_BUS_TYPE_CODE) {
```

```

51     Serial.println("ID[1] is Unit ChainBus\n");
52     // Configure GPIO1 as NVIC (external interrupt) mode, rising edge trigger
53     chain_status = M5Chain.setChainBusNvicMode(devices_list->devices[0].id, CHAIN_GPIO_PIN_1, CHAIN_GPIO_MODE_IT_RISING, &operation_status);
54     CHAIN_GPIO_MODE_IT_RISING, &operation_status);
55     if (chain_status == CHAIN_OK && operation_status == 1) {
56         Serial.printf("ChainBus ID[%d] set gpio1 nvic mode RISING success \r\n", devices_list->devices[0].id, chain_status, operation_status);
57     } else {
58         Serial.printf("ChainBus ID[%d] set gpio1 nvic mode RISING fail, chain_status:%d operation_status:%d\n", devices_list->devices[0].id, chain_status, operation_status);
59     }
60 }
61
62 // Configure GPIO2 as NVIC (external interrupt) mode, rising and falling edge trigger
63 chain_status = M5Chain.setChainBusNvicMode(devices_list->devices[0].id, CHAIN_GPIO_PIN_2, CHAIN_GPIO_MODE_IT_RISING_FALLING, &operation_status);
64 CHAIN_GPIO_MODE_IT_RISING_FALLING, &operation_status);
65 if (chain_status == CHAIN_OK && operation_status == 1) {
66     Serial.printf("ChainBus ID[%d] set gpio2 nvic mode RISING_FALLING success \r\n", devices_list->devices[0].id, chain_status, operation_status);
67 } else {
68     Serial.printf("ChainBus ID[%d] set gpio2 mode RISING_FALLING fail, chain_status:%d operation_status:%d\n", devices_list->devices[0].id, chain_status, operation_status);
69 }
70 }
71 M5Chain.getChainBusWorkMode(devices_list->devices[0].id, &gpio1, &gpio2);
72 printWorkStatus("GPIO_1", (work_status_t)gpio1);
73 printWorkStatus("GPIO_2", (work_status_t)gpio2);
74 delay(500);
75 } else {
76     Serial.println("ID[1] is NOT Unit ChainBus\n");
77     return;
78 }
79
80 delay(5);
81 }
82
83 void loop()
84 {
85     if (devices_list) {
86         if (devices_list->devices[0].device_type == UNIT_CHAIN_BUS_TYPE_CODE) {
87             // Check whether GPIO1 or GPIO2 is configured as NVIC mode
88             if (gpio1 == CHAIN_NVIC_WORK_STATUS || gpio2 == CHAIN_NVIC_WORK_STATUS) {
89                 uint16_t nvic_status = 0;
90                 while (M5Chain.getChainBusNvicTriggerStatus(devices_list->devices[0].id, &nvic_status)) {
91                     Serial.println("*****");
92                     Serial.printf("ChainBus ID[%d] nvic status: 0x%04x \r\n", devices_list->devices[0].id, nvic_status);
93                 }
94             }
95         } else {
96             Serial.printf("ChainBus ID[%d] get work mode fail, chain_status:%d operation_status:%d\n", devices_list->devices[0].id, chain_status, operation_status);
97         }
98     }
99 }
100
101 delay(40);
102 }
103

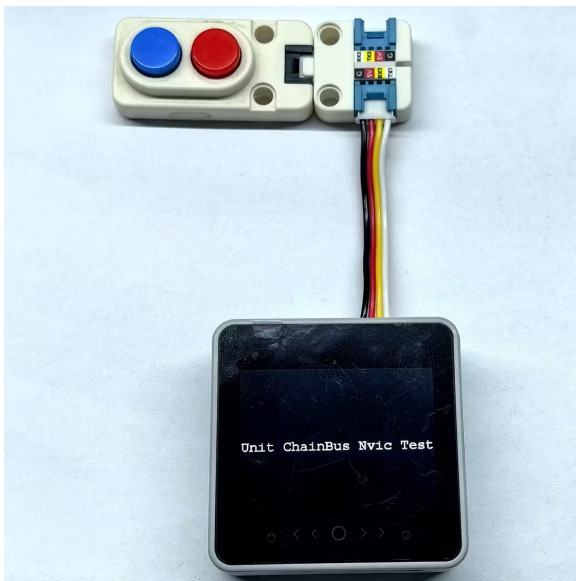
```

```

104 void printWorkStatus(String gpio_num, work_status_t gpio_status)
105 {
106     Serial.print(" >>> " + gpio_num + ": ");
107     switch (gpio_status) {
108         case CHAIN_NOT_WORK_STATUS:
109             Serial.println("Not configured working status");
110             break;
111         case CHAIN_OUTPUT_WORK_STATUS:
112             Serial.println("Output status");
113             break;
114         case CHAIN_INPUT_WORK_STATUS:
115             Serial.println("Input status");
116             break;
117         case CHAIN_NVIC_WORK_STATUS:
118             Serial.println("External interrupt working status");
119             break;
120         case CHAIN_ADC_WORK_STATUS:
121             Serial.println("ADC working status");
122             break;
123         case CHAIN_I2C_WORK_STATUS:
124             Serial.println("I2C working status");
125             break;
126         default:
127             Serial.println("Unrecognized work status");
128             break;
129     }
130 }

```

设备上电后，串口监视器会输出连接设备的信息，并通过 Unit ChainBus 的接口读取连接在其上的 Unit Dual Button 按钮的中断触发状态。按动蓝色按键，会触发上升沿中断，串口返回一次相应信息；按动红色按键，会触发上升沿和下降沿中断，串口返回两次相应信息。返回的十六进制值中，低字节表示 GPIO 触发中断前的电平状态，高字节表示触发中断的 GPIO 编号。



串口监视器输出示例：

```
Unit ChainBus Nvic Test
devices is connected
devices list get success
ID[1] is Unit ChainBus

ChainBus ID[1] set gpio1 nvic mode RISING success
ChainBus ID[1] set gpio2 nvic mode RISING_FALLING success
>>> GPIO_1: External interrupt working status
>>> GPIO_2: External interrupt working status
*****
ChainBus ID[1] nvic status: 0x0100
*****
ChainBus ID[1] nvic status: 0x0201
*****
ChainBus ID[1] nvic status: 0x0200
```

4.5 ADC 模式

```
1  #include "M5Chain.h"
2  #include "M5Unified.h"
3
4  #define TXD_PIN GPIO_NUM_17 // Tx
5  #define RXD_PIN GPIO_NUM_18 // Rx
6
7  Chain M5Chain;
8
9  device_list_t *devices_list = NULL;
10 chain_status_t chain_status;
11 uint16_t device_nums = 0;
12 uint8_t operation_status = 0;
13 work_status_t gpio1;
14 work_status_t gpio2;
15
16 void printWorkStatus(String gpio_num, work_status_t gpio_status);
17
18 void setup()
19 {
20     M5.begin();
21     M5.Display.clear();
22     M5.Display.setFont(&FreeMonoBold12pt7b);
23     M5.Display.setTextDatum(middle_center);
24     M5.Display.drawCenterString("Unit ChainBus Test", 160, 120);
25
26     Serial.begin(115200);
27     Serial.println("Unit ChainBus Test");
28     M5Chain.begin(&Serial2, 115200, RXD_PIN, TXD_PIN);
29
30     if (M5Chain.isDeviceConnected()) {
31         Serial.println("devices is connected");
32         chain_status = M5Chain.getDeviceNum(&device_nums);
33         if (chain_status == CHAIN_OK) {
34             devices_list = (device_list_t *)malloc(sizeof(device_list_t));
35             devices_list->count = device_nums;
36             devices_list->devices = (device_info_t *)malloc(sizeof(device_info_t) * device_nums);
37             if (M5Chain.getDeviceList(devices_list)) {
38                 Serial.println("devices list get success");
39             } else {
40                 Serial.println("devices list get failed");
41             }
42         } else {
43             Serial.printf("error status:%d \r\n", chain_status);
44             Serial.printf("devices num get failed.\r\n");
45         }
46     } else {
47         Serial.println("devices is not connected.");
48     }
49
50     if (devices_list->devices[0].device_type == UNIT_CHAIN_BUS_TYPE_CODE) {
```

```

51     Serial.println("ID[1] is Unit ChainBus\n");
52
53     // Configure GPIO1 as ADC input mode
54     chain_status = M5Chain.setChainBusAdcMode(devices_list->devices[0].id, CHAIN_GPIO_PIN_1, &op
55     if (chain_status == CHAIN_OK && operation_status == 1) {
56         Serial.printf("ChainBus ID[%d] set GPIO1 adc mode success \r\n", devices_list->devices[0
57     } else {
58         Serial.printf("ChainBus ID[%d] set GPIO1 adc mode fail, chain_status:%d operation_status:
59             devices_list->devices[0].id, chain_status, operation_status);
60     }
61
62     M5Chain.getChainBusWorkMode(devices_list->devices[0].id, &gpio1, &gpio2);
63     printWorkStatus("GPIO_1", (work_status_t)gpio1);
64     printWorkStatus("GPIO_2", (work_status_t)gpio2);
65     delay(500);
66 } else {
67     Serial.println("ID[1] is NOT Unit ChainBus\n");
68     return;
69 }
70
71 delay(5);
72 }
73
74 uint16_t last_adc_value = 0;
75
76 void loop()
77 {
78     uint16_t adc_value;
79     chain_status = M5Chain.getChainBusAdcValue(devices_list->devices[0].id, CHAIN_GPIO_PIN_1, &adc_v
80     if (chain_status == CHAIN_OK && operation_status == 1) {
81         if (abs(adc_value - last_adc_value) > 4) { // Print ADC value only when the change exceeds t
82             Serial.printf("\nADC value: %d \r\n", adc_value);
83             last_adc_value = adc_value;
84         }
85     } else {
86         Serial.printf("get ADC value failed, chain_status:%d operation_status:%d \r\n", chain_statu
87     }
88
89     delay(100);
90 }
91
92 void printWorkStatus(String gpio_num, work_status_t gpio_status)
93 {
94     Serial.print(" >>> " + gpio_num + ": ");
95     switch (gpio_status) {
96         case CHAIN_NOT_WORK_STATUS:
97             Serial.println("Not configured working status");
98             break;
99         case CHAIN_OUTPUT_WORK_STATUS:
100             Serial.println("Output status");
101             break;
102         case CHAIN_INPUT_WORK_STATUS:
103             Serial.println("Input status");

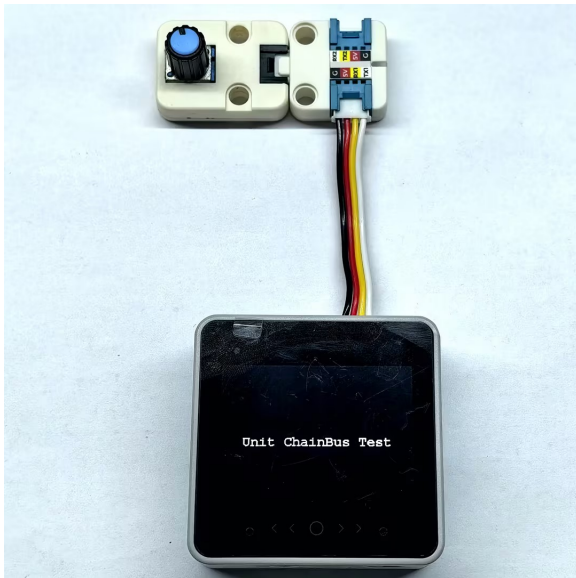
```

```

104         break;
105     case CHAIN_NVIC_WORK_STATUS:
106         Serial.println("External interrupt working status");
107         break;
108     case CHAIN_ADC_WORK_STATUS:
109         Serial.println("ADC working status");
110         break;
111     case CHAIN_I2C_WORK_STATUS:
112         Serial.println("I2C working status");
113         break;
114     default:
115         Serial.println("Unrecognized work status");
116         break;
117 }
118 }

```

设备上电后，串口监视器会输出连接设备的信息，并通过 Unit ChainBus 的接口读取连接在其上的 Unit Angle 的 ADC 数值。



串口监视器输出示例：

```

Unit ChainBus Test
devices is connected
devices list get success
ID[1] is Unit ChainBus

ChainBus ID[1] set GPIO1 adc mode success
>>> GPIO_1: ADC working status
>>> GPIO_2: Not configured working status

ADC value: 4095

```