



DesignNews

Designing, Building and Coding Custom Raspberry Pi RP2040 Arduino Devices

Day 4:

The RP2040, Arduino and Wi-Fi

Sponsored by

DigiKey



Webinar Logistics

- Turn on your system sound to hear the streaming presentation.
- If you have technical problems, click “Help” or submit a question asking for assistance.
- Participate in ‘Attendee Chat’ by maximizing the chat widget in your dock.

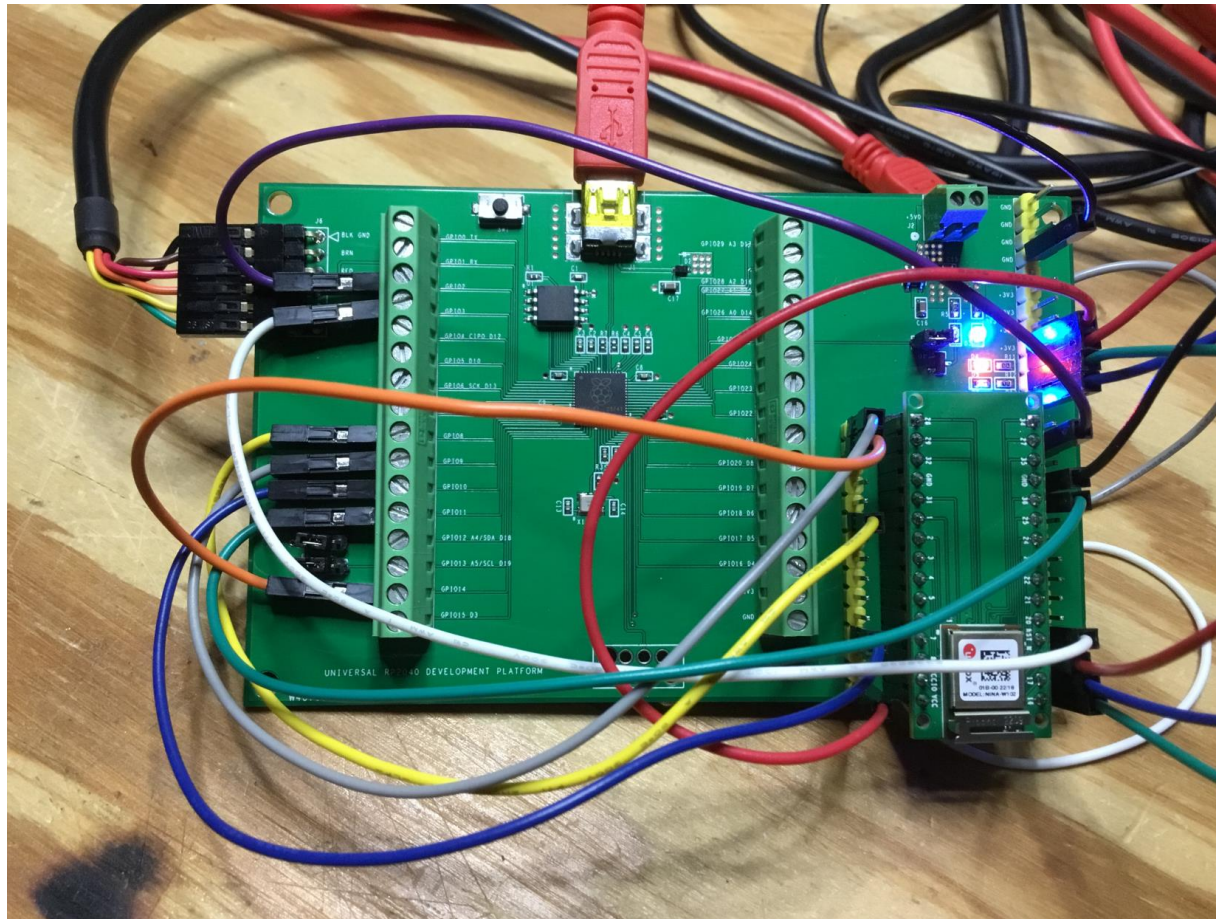


Fred Eady

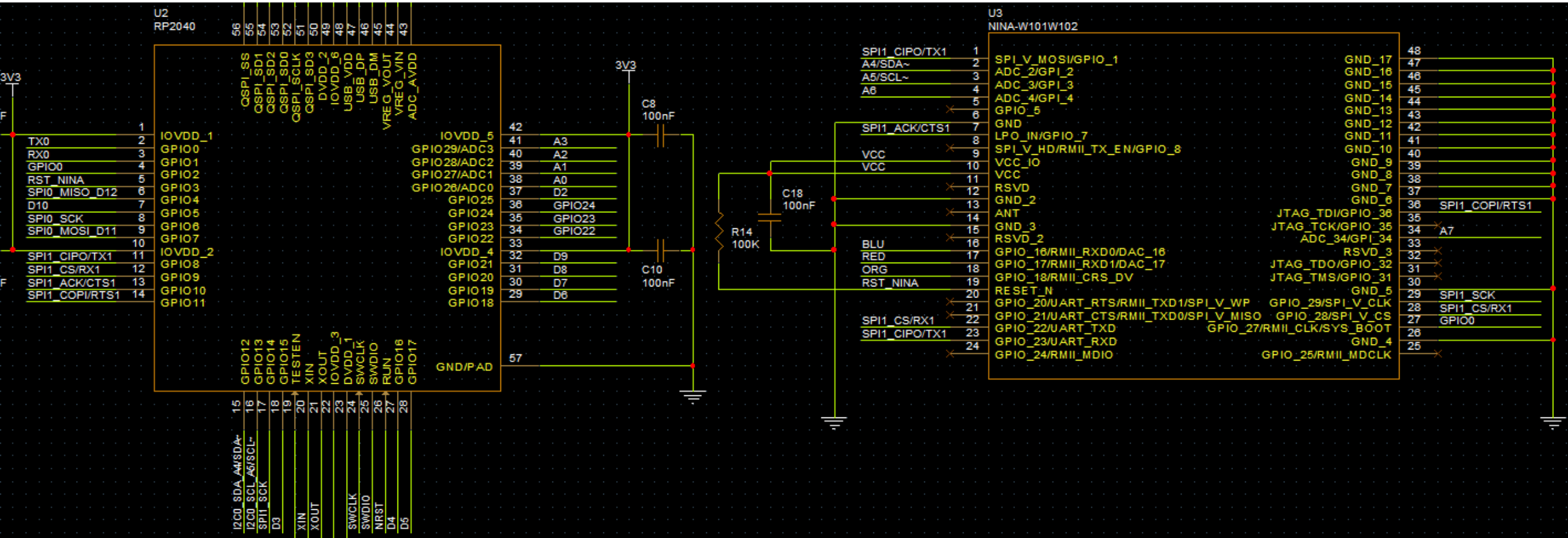
Visit 'Lecturer Profile' in your console for more details.

AGENDA

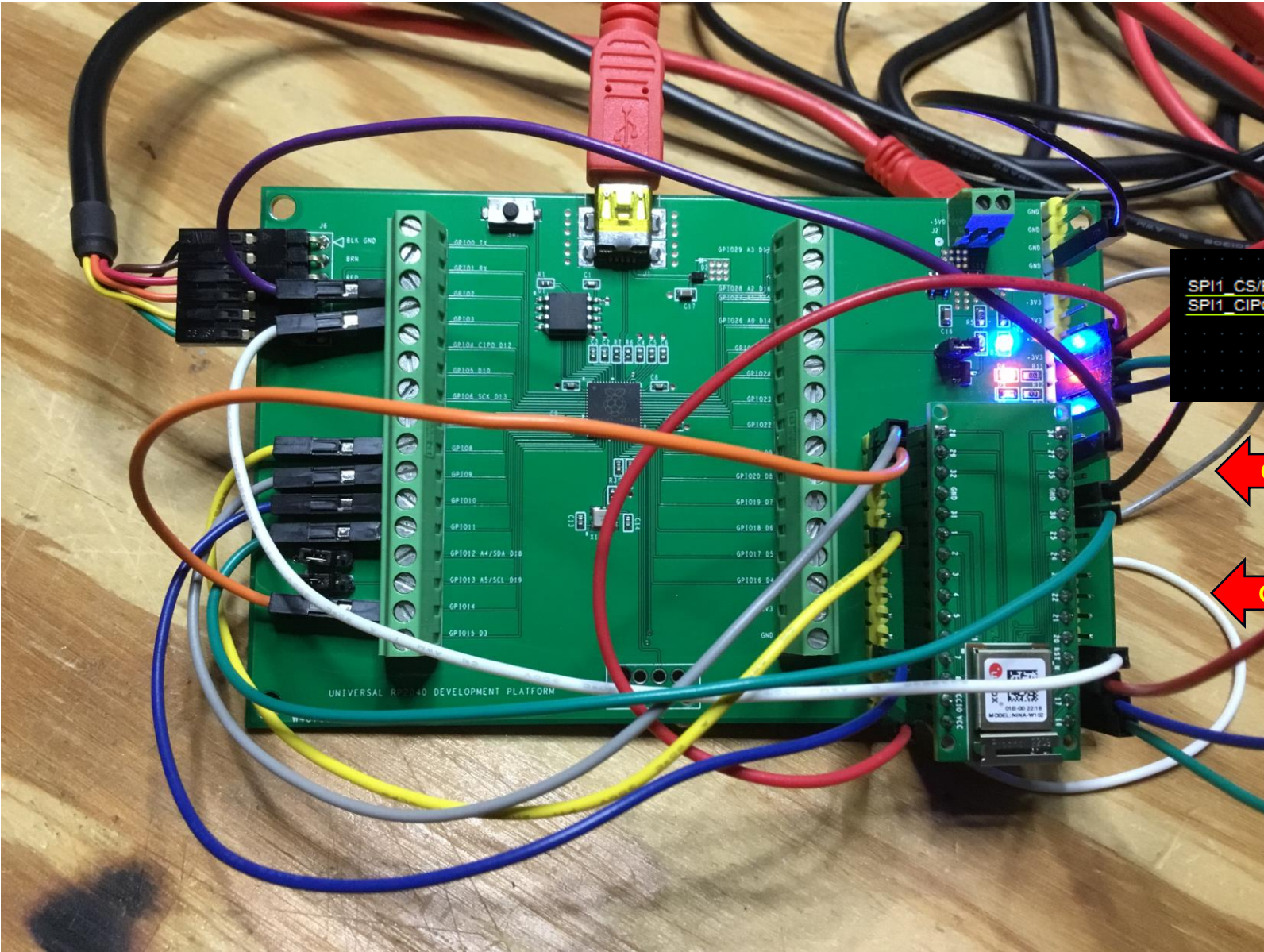
- **RP2040 Arduino Wi-Fi Hardware**
- **u-blox Wi-Fi Powered by Arduino**
- **Wi-Fi Remote Control**



RP2040 Arduino Wi-Fi Hardware Design



RP2040 Arduino Wi-Fi Hardware Design



SPI1_CS/RX1	20	GPIO_20/UART_RTS/R
SPI1_CIPO/TX1	22	GPIO_21/UART_CTS/R
	23	GPIO_22/UART_TXD
	24	GPIO_23/UART_RXD
		GPIO_24/RMII_MDIO

OPTIONAL FOR WI-FI ←

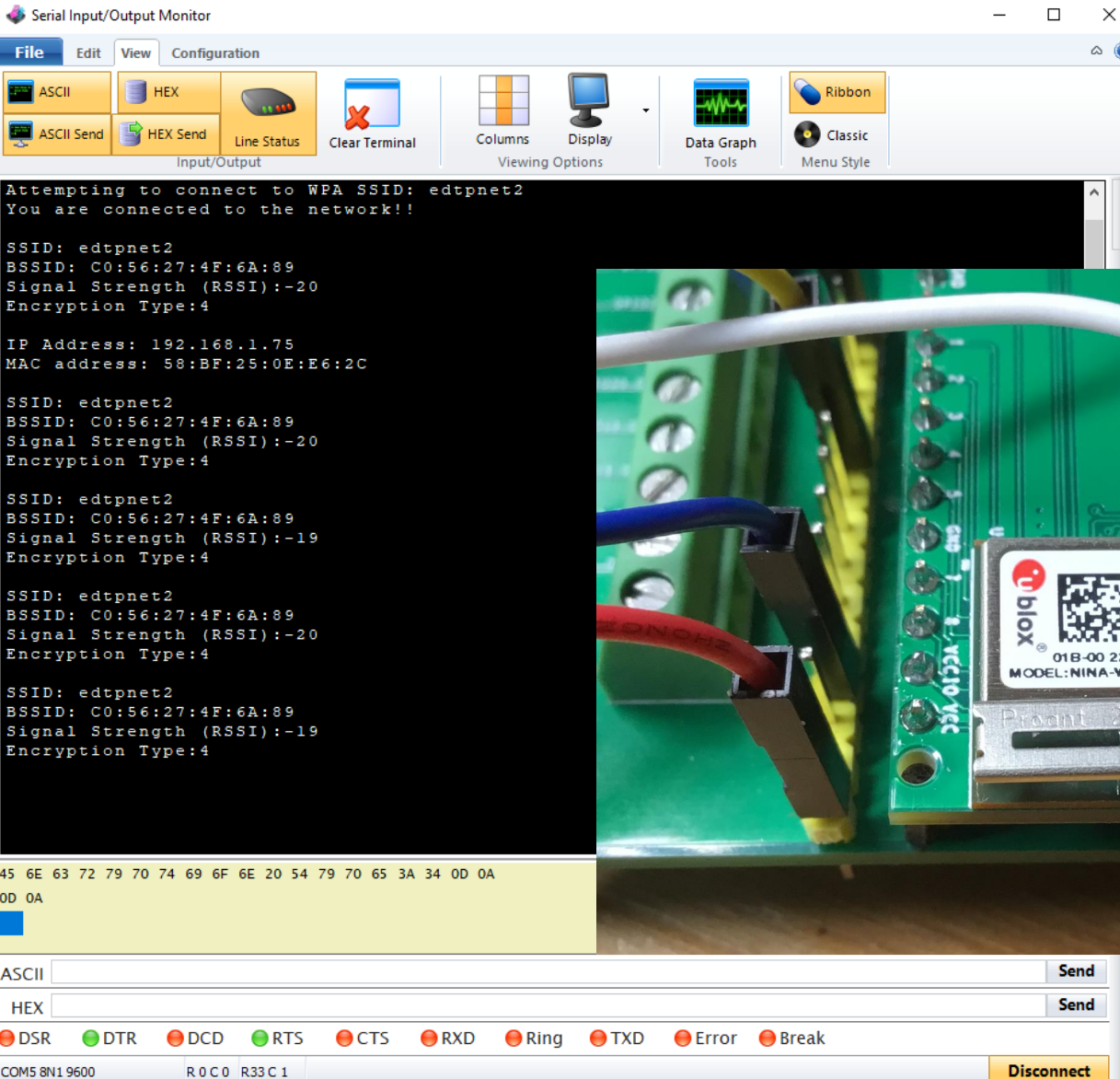
OPTIONAL FOR WI-FI ←

Arduino Wi-Fi First Contact – setup()

```

11 #include <SPI.h>
12 #include <WiFiNINA.h>
13 #include "arduino_secrets.h"
14 // enter your sensitive data in arduino_secrets.h
15 char ssid[] = SECRET_SSID; // your network SSID
16 char pass[] = SECRET_PASS; // your network password
17 int status = WL_IDLE_STATUS; // the WiFi radio's status
18
19 void setup() {
20 //Initialize serial and wait for port to open:
21 Serial1.begin(9600);
22 while(!Serial1);
23 // check for the WiFi module:
24 if (WiFi.status() == WL_NO_MODULE) {
25 Serial1.println("Communication with WiFi module failed!");
26 // spin here and don't continue
27 while (true);
28 }
29 // check NINA firmware version
30 String fv = WiFi.firmwareVersion();
31 if (fv < WIFI_FIRMWARE_LATEST_VERSION) {
32 Serial1.println("Please upgrade the firmware");
33 }
34 // attempt to connect to WiFi network:
35 while (status != WL_CONNECTED) {
36 Serial1.print("Attempting to connect to WPA SSID: ");
37 Serial1.println(ssid);
38 // Connect to WPA/WPA2 network:
39 status = WiFi.begin(ssid, pass);
40 // wait 10 seconds for connection:
41 delay(10000);
42 }
43 // you're connected now, so print out the data:
44 Serial1.println("You are connected to the network!!\r\n");
45 printCurrentNet();
46 printWifiData();
47 Serial1.println();
48 }

```



Serial Input/Output Monitor

File Edit View Configuration

ASCII HEX Line Status Clear Terminal Columns Display Data Graph Ribbon ASCII Send HEX Send Input/Output Viewing Options Tools Menu Style

Attempting to connect to WPA SSID: edtpnet2
You are connected to the network!!

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-20
Encryption Type:4

IP Address: 192.168.1.75
MAC address: 58:BF:25:0E:E6:2C

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-20
Encryption Type:4

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-19
Encryption Type:4

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-20
Encryption Type:4

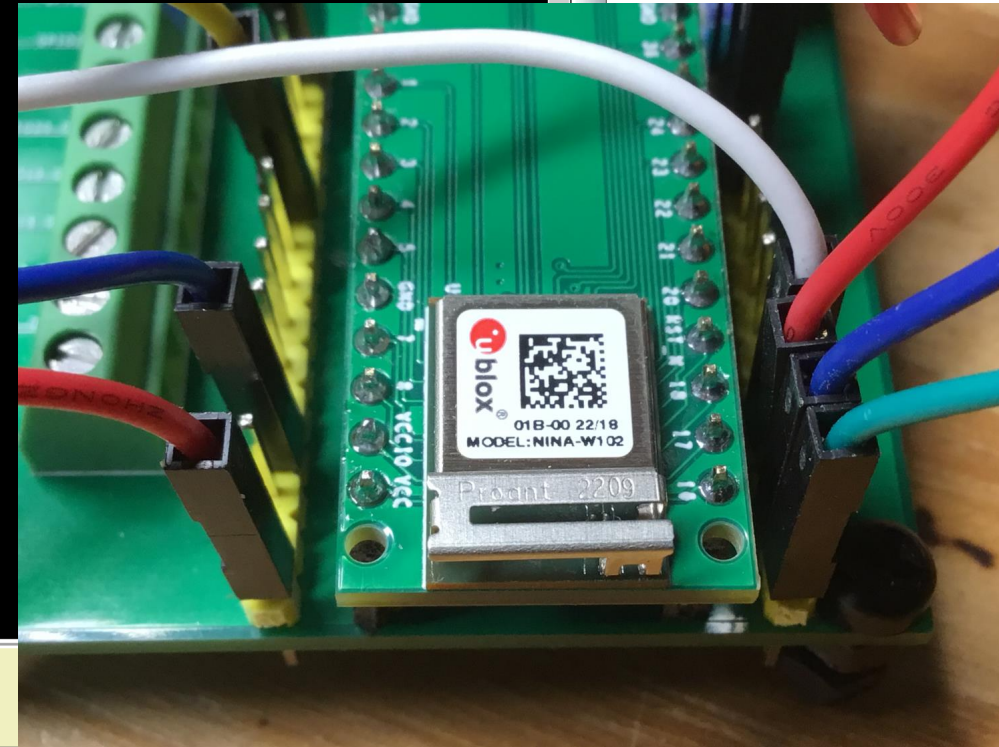
SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-19
Encryption Type:4

45 6E 63 72 79 70 74 69 6F 6E 20 54 79 70 65 3A 34 0D 0A
0D 0A

ASCII Send
HEX Send

DSR DTR DCD RTS CTS RXD Ring TXD Error Break

COM5 8N1 9600 R 0 C 0 R33 C 1 Disconnect

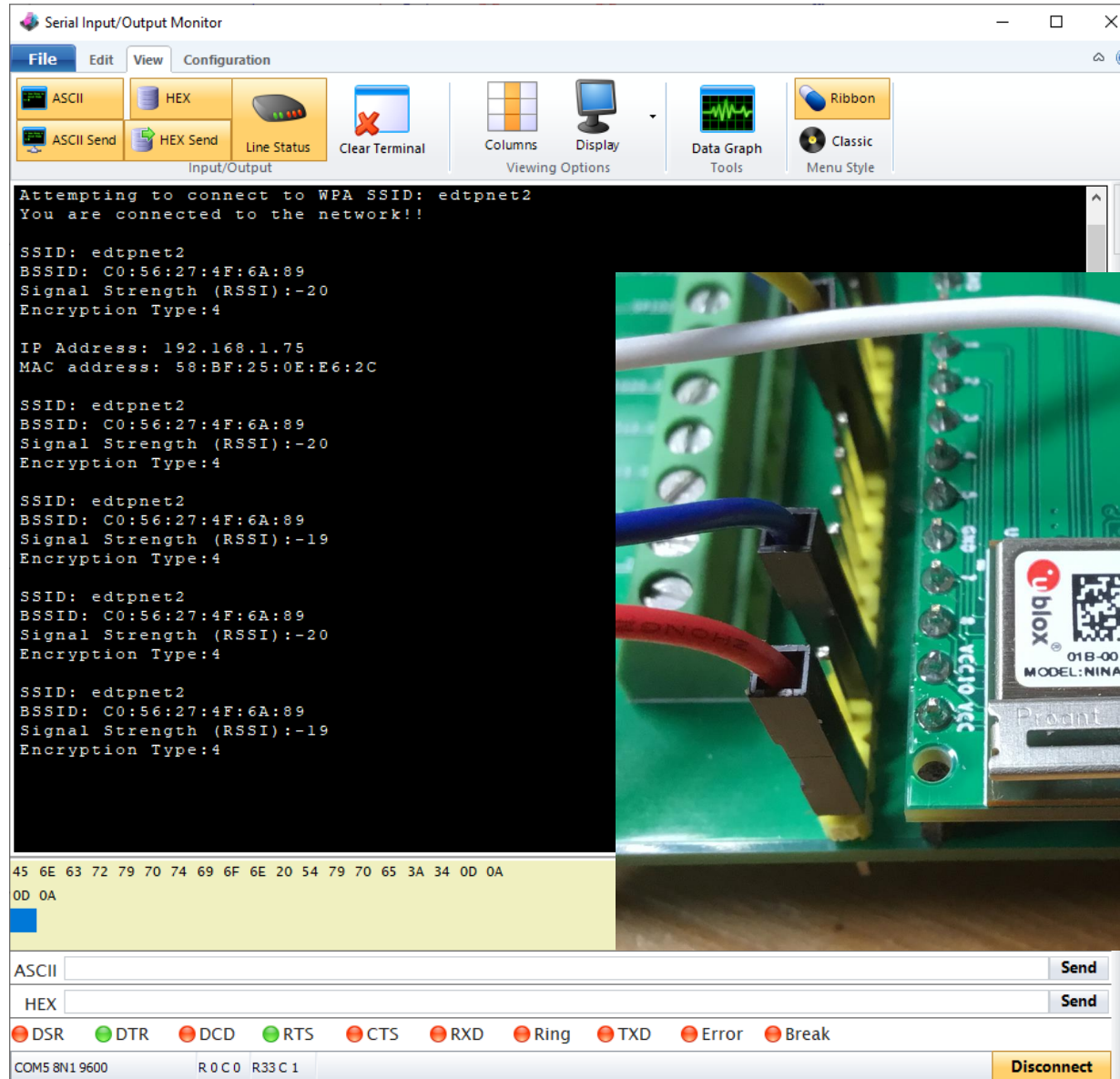


Arduino Wi-Fi First Contact – setup() functions

```

78 void printWifiData() {
79     // print your board's IP address:
80     IPAddress ip = WiFi.localIP();
81     Serial1.print("IP Address: ");
82     Serial1.println(ip);
83
84     // print your MAC address:
85     byte mac[6];
86     WiFi.macAddress(mac);
87     Serial1.print("MAC address: ");
88     printMacAddress(mac);
89 }
90
91 void printMacAddress(byte mac[]) {
92     for (int i = 5; i >= 0; i--) {
93         if (mac[i] < 16) {
94             Serial1.print("0");
95         }
96         Serial1.print(mac[i], HEX);
97         if (i > 0) {
98             Serial1.print(":");
99         }
100     }
101     Serial1.println();
102 }

```



Serial Input/Output Monitor

File Edit View Configuration

ASCII HEX Line Status Clear Terminal Columns Display Data Graph Ribbon

ASCII Send HEX Send Input/Output Viewing Options Tools Menu Style

Attempting to connect to WPA SSID: edtpnet2
You are connected to the network!!

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-20
Encryption Type:4

IP Address: 192.168.1.75
MAC address: 58:BF:25:0E:E6:2C

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-20
Encryption Type:4

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-19
Encryption Type:4

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-20
Encryption Type:4

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-19
Encryption Type:4

45 6E 63 72 79 70 74 69 6F 6E 20 54 79 70 65 3A 34 0D 0A
0D 0A

ASCII Send

HEX Send

DSR DTR DCD RTS CTS RXD Ring TXD Error Break

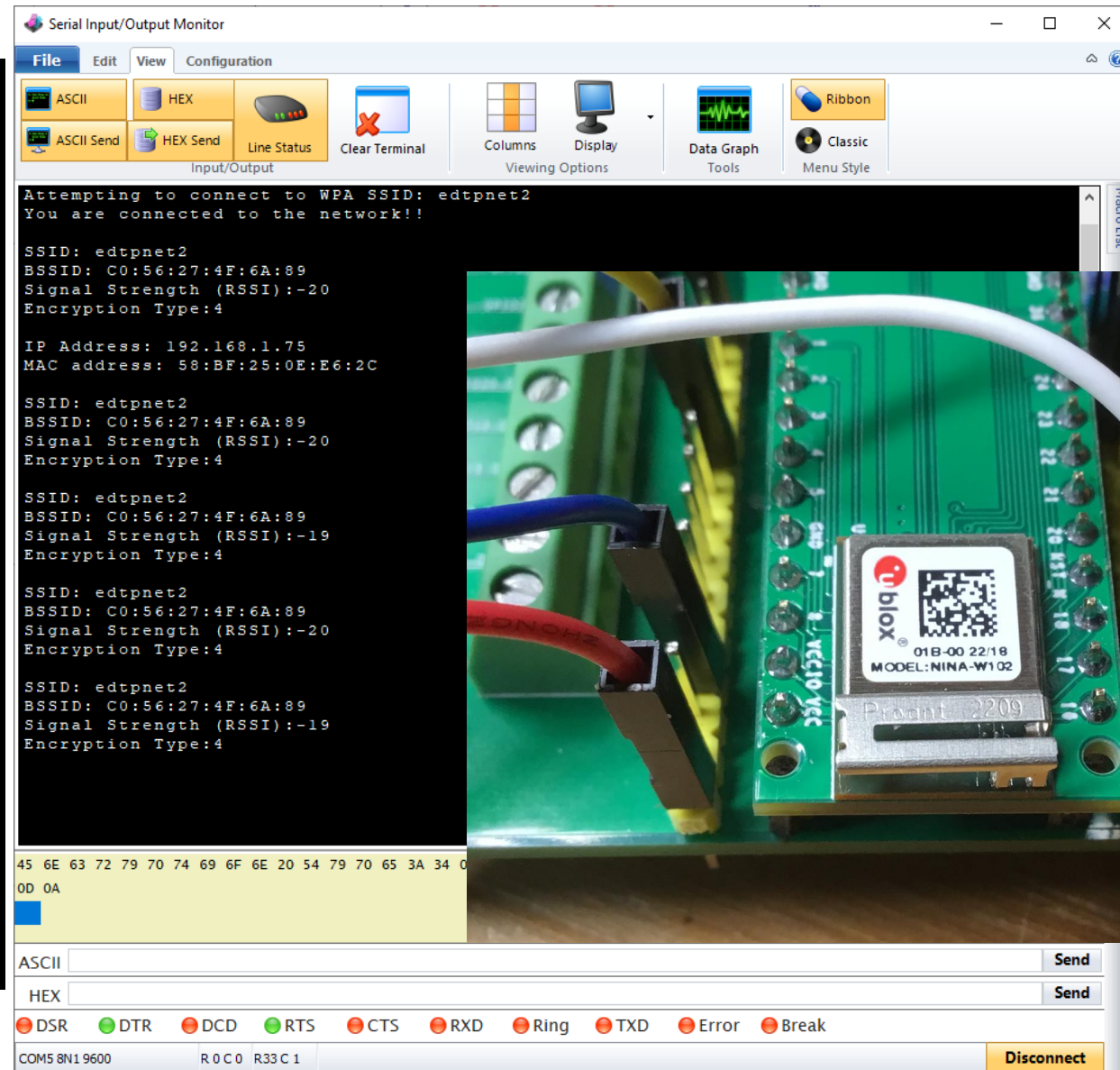
COM5 8N19600 R 0 C 0 R 33 C 1 Disconnect

Arduino Wi-Fi First Contact – loop()

```

50 void loop() {
51     // check the network connection once every 10 seconds:
52     delay(10000);
53     printCurrentNet();
54 }
55
56 void printCurrentNet() {
57     // print the SSID of the network you're attached to:
58     Serial1.print("SSID: ");
59     Serial1.println(WiFi.SSID());
60     // print the MAC address of the router you're attached to:
61     byte bssid[6];
62     WiFi.BSSID(bssid);
63     Serial1.print("BSSID: ");
64     printMacAddress(bssid);
65
66     // print the received signal strength:
67     long rssi = WiFi.RSSI();
68     Serial1.print("Signal Strength (RSSI):");
69     Serial1.println(rssi);
70
71     // print the encryption type:
72     byte encryption = WiFi.encryptionType();
73     Serial1.print("Encryption Type:");
74     Serial1.println(encryption, HEX);
75     Serial1.println();
76 }

```



The screenshot shows the Serial Input/Output Monitor window with the following output:

```

Attempting to connect to WPA SSID: edtpnet2
You are connected to the network!!

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-20
Encryption Type:4

IP Address: 192.168.1.75
MAC address: 58:BF:25:0E:E6:2C

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-20
Encryption Type:4

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-19
Encryption Type:4

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-20
Encryption Type:4

SSID: edtpnet2
BSSID: C0:56:27:4F:6A:89
Signal Strength (RSSI):-19
Encryption Type:4

```

The photograph shows a u-blox Wi-Fi module (Model: NINA-W102) mounted on a green PCB. The module is connected to a terminal block with several colored wires (red, blue, yellow, green, black). The module has a QR code and the text "u-blox 01B-00 22/18 MODEL: NINA-W102" and "Proant 2209".

The Serial Input/Output Monitor window also shows the following settings:

- File Edit View Configuration
- Input/Output: ASCII, HEX, ASCII Send, HEX Send, Line Status, Clear Terminal
- Viewing Options: Columns, Display
- Tools: Data Graph
- Menu Style: Ribbon, Classic
- COM5 8N1 9600 R 0 C 0 R33 C 1
- Buttons: Send, Disconnect
- Legend: DSR (red), DTR (green), DCD (red), RTS (green), CTS (red), RXD (red), Ring (red), TXD (red), Error (red), Break (red)

Driving NINA GPIO Pins – nina_pins.cpp/nina_pins.h

```

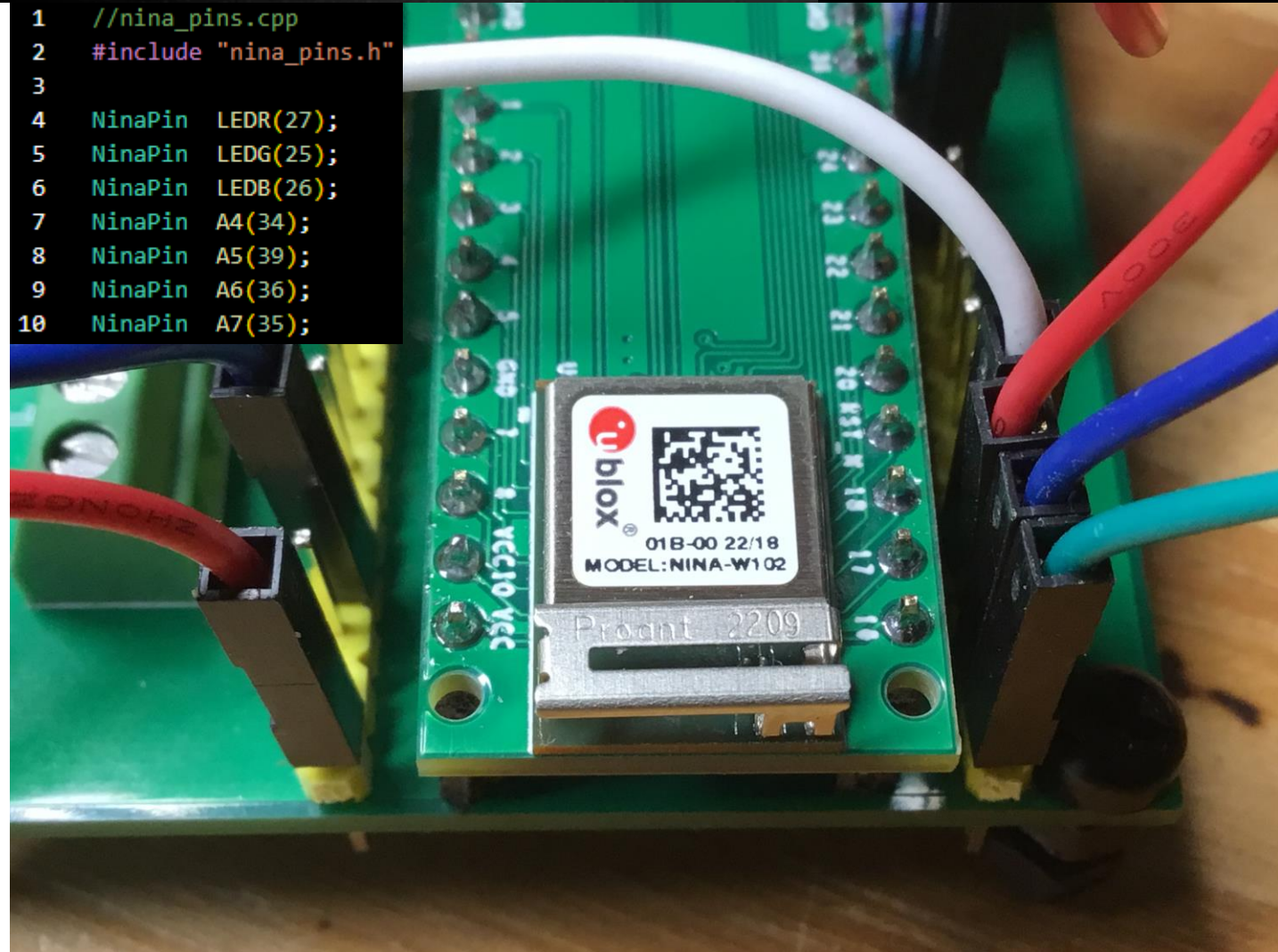
26 class NinaPin {
27 public:
28     NinaPin(int _pin) : pin(_pin) {};
29     int get() {
30         return pin;
31     };
32     int analogReadResolution() {
33         return getAnalogReadResolution();
34     };
35     bool operator==(NinaPin const & other) const {
36         return pin == other.pin;
37     }
38     //operator int() = delete;
39     __attribute__((error("Change me to a #define"))) operator int();
40 private:
41     int pin;
42 };
43
44 extern NinaPin LEDR;
45 extern NinaPin LEDG;
46 extern NinaPin LEDB;
47 extern NinaPin A4;
48 extern NinaPin A5;
49 extern NinaPin A6;
50 extern NinaPin A7;
51
52 #define NINA_PINS_AS_CLASS
53
54 /*****
55  * FUNCTION DECLARATION
56  *****/
57
58 void NINA_ATTRIBUTE pinMode (NinaPin pin, PinMode mode);
59 PinStatus NINA_ATTRIBUTE digitalRead (NinaPin pin);
60 void NINA_ATTRIBUTE digitalWrite(NinaPin pin, PinStatus value);
61 int NINA_ATTRIBUTE analogRead (NinaPin pin);
62 void NINA_ATTRIBUTE analogWrite (NinaPin pin, int value);
63
64 #undef NINA_ATTRIBUTE

```

```

1 //nina_pins.cpp
2 #include "nina_pins.h"
3
4 NinaPin LEDR(27);
5 NinaPin LEDG(25);
6 NinaPin LEDB(26);
7 NinaPin A4(34);
8 NinaPin A5(39);
9 NinaPin A6(36);
10 NinaPin A7(35);

```



Driving NINA GPIO Pins

```

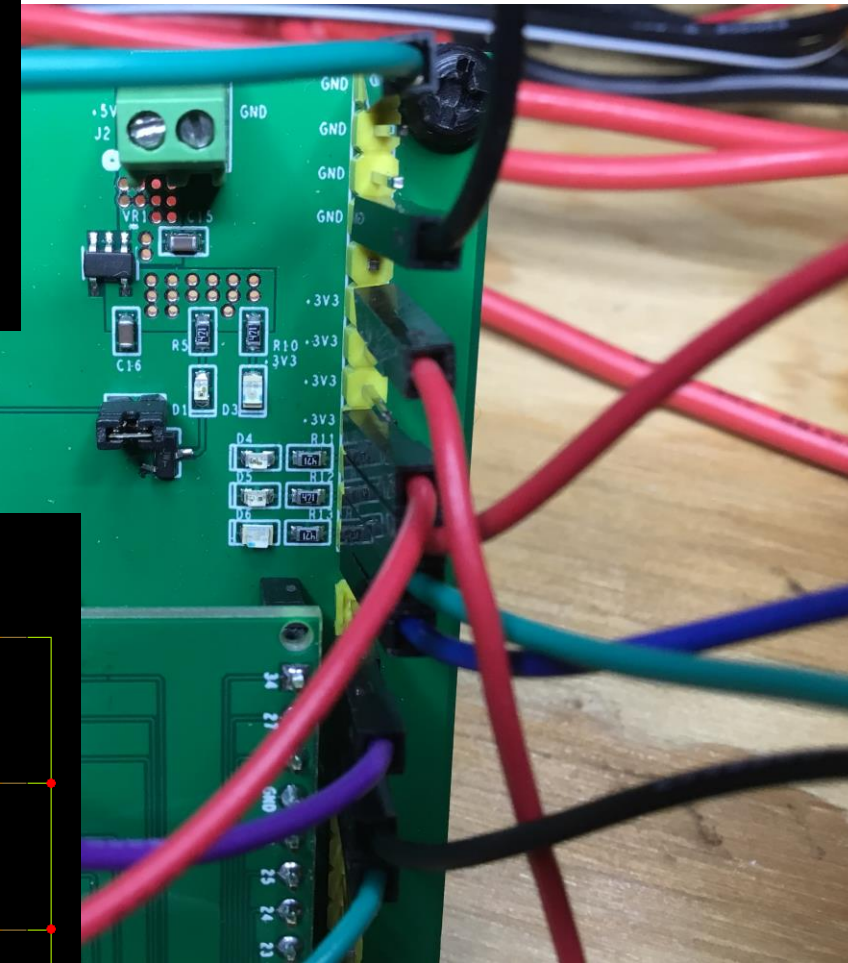
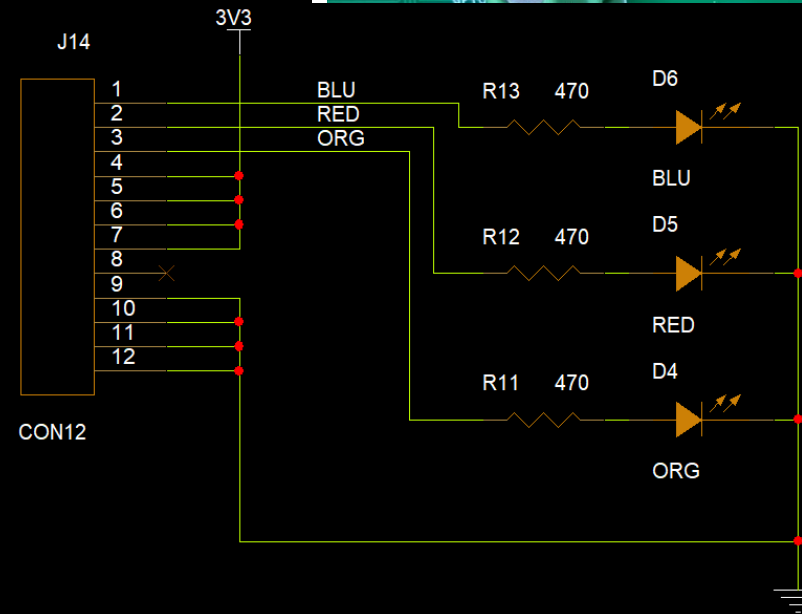
4  #include <SPI.h>
5  #include <WiFiNINA.h>
6  #include <WiFiUdp.h>
7  #include <utility/wifi_drv.h>
8  #include "arduino_secrets.h"
9
10 // LEDs on Universal RP2040 Development Platform
11 // reference nina_pins.cpp and nina_pins.h
12 #define ORG 27 // associates to -> NinaPin LEDR(27)
13 #define RED 25 // associates to -> NinaPin LEDG(25)
14 #define BLU 26 // associates to -> NinaPin LEDB(26)
15
16 uint8_t i;
17 int status = WL_IDLE_STATUS;
18 // enter your sensitive data in arduino_secrets.h
19 char ssid[] = SECRET_SSID; // your network SSID (name)
20 char pass[] = SECRET_PASS; // your network password (use for WPA, or use as key for WEP)
21 int keyIndex = 0; // your network key index number (needed only for WEP)
22 unsigned int localPort = 8088; // local port used to listen
23 char packetBuffer[256]; //buffer to hold incoming packet
24 char ackBuffer[64]; // contains an acknowledge string
25 WiFiUDP Udp; //WiFiUDP is a class
26
27 void setup() {
28   WiFiDrv::pinMode(ORG,OUTPUT);
29   WiFiDrv::pinMode(RED,OUTPUT);
30   WiFiDrv::pinMode(BLU,OUTPUT);
31   WiFiDrv::digitalWrite(ORG,LOW);
32   WiFiDrv::digitalWrite(RED,LOW);
33   WiFiDrv::digitalWrite(BLU,LOW);
34
35   pinMode(LED_BUILTIN,OUTPUT);
36   digitalWrite(LED_BUILTIN,HIGH);
37   delay(1000);
38   digitalWrite(LED_BUILTIN,LOW);
39   //Initialize Serial1 and wait for port to open:
40   Serial1.begin(9600);
41   while(!Serial1);

```

```

1  //nina_pins.cpp
2  #include "nina_pins.h"
3
4  NinaPin LEDR(27);
5  NinaPin LEDG(25);
6  NinaPin LEDB(26);
7  NinaPin A4(34);
8  NinaPin A5(39);
9  NinaPin A6(36);
10 NinaPin A7(35);

```



Driving NINA GPIO Pins

```

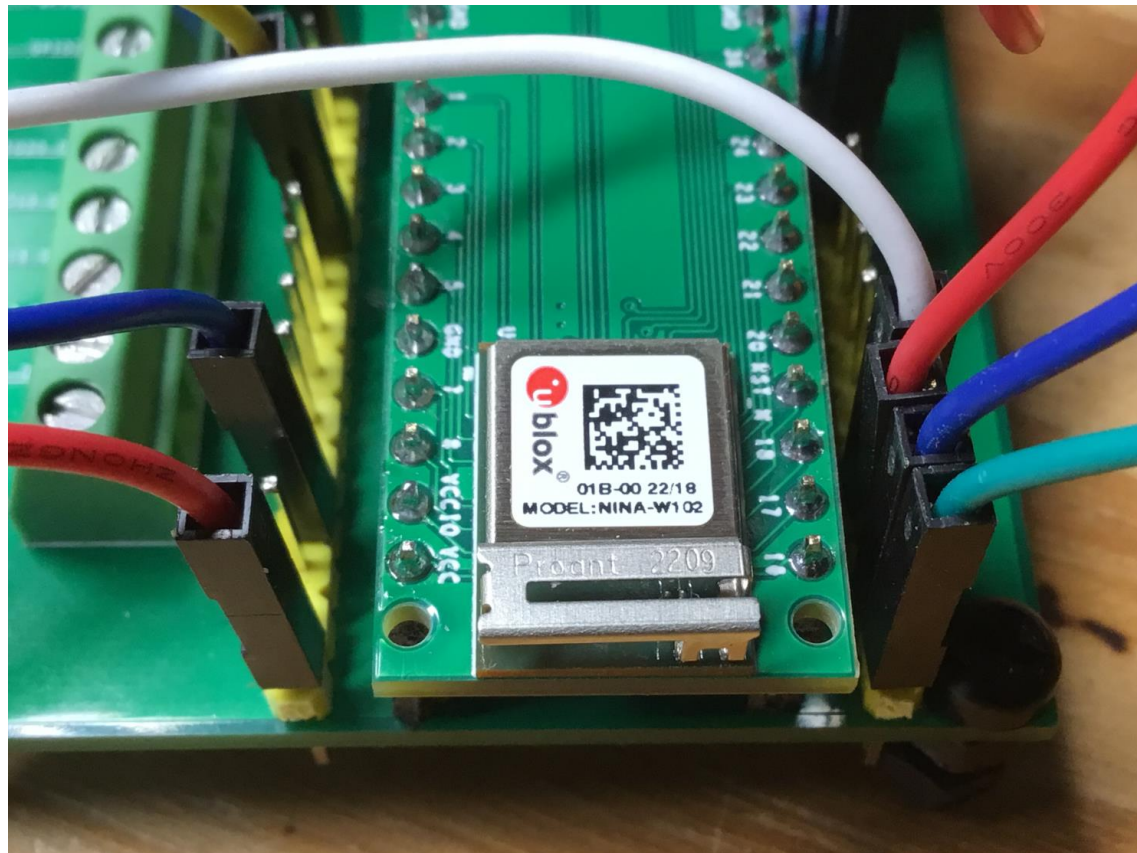
27 void setup() {
28     WiFiDrv::pinMode(ORG,OUTPUT);
29     WiFiDrv::pinMode(RED,OUTPUT);
30     WiFiDrv::pinMode(BLU,OUTPUT);
31     WiFiDrv::digitalWrite(ORG,LOW);
32     WiFiDrv::digitalWrite(RED,LOW);
33     WiFiDrv::digitalWrite(BLU,LOW);

```

```

1036 void WiFiDrv::pinMode(uint8_t pin, uint8_t mode)
1037 {
1038     WAIT_FOR_SLAVE_SELECT();
1039     // Send Command
1040     SpiDrv::sendCmd(SET_PIN_MODE, PARAM_NUMS_2);
1041     SpiDrv::sendParam((uint8_t*)&pin, 1, NO_LAST_PARAM);
1042     SpiDrv::sendParam((uint8_t*)&mode, 1, LAST_PARAM);
1043
1044     // pad to multiple of 4
1045     SpiDrv::readChar();
1046
1047     SpiDrv::spiSlaveDeselect();
1048     //Wait the reply elaboration
1049     SpiDrv::waitForSlaveReady();
1050     SpiDrv::spiSlaveSelect();
1051
1052     // Wait for reply
1053     uint8_t _data = 0;
1054     uint8_t _dataLen = 0;
1055     if (!SpiDrv::waitResponseCmd(SET_PIN_MODE, PARAM_NUMS_1, &_data, &_dataLen))
1056     {
1057         WARN("error waitResponse");
1058         _data = WL_FAILURE;
1059     }
1060     SpiDrv::spiSlaveDeselect();
1061 }

```



Driving NINA GPIO Pins

```

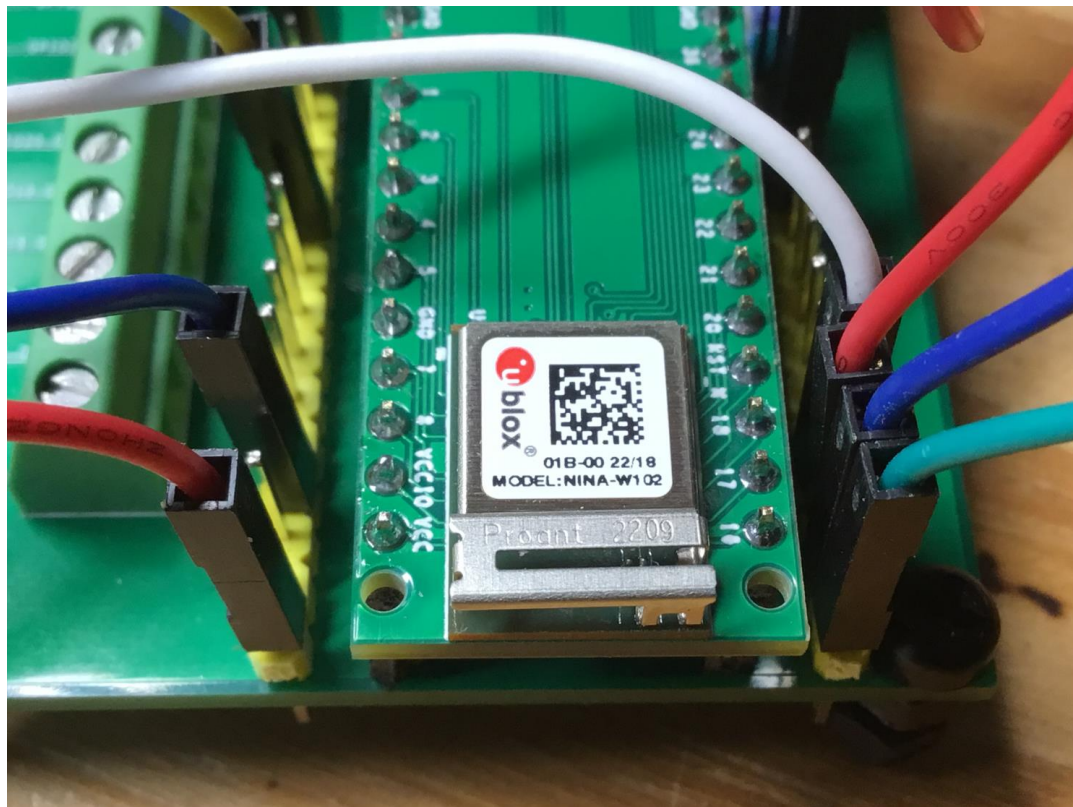
27 void setup() {
28     WiFiDrv::pinMode(ORG,OUTPUT);
29     WiFiDrv::pinMode(RED,OUTPUT);
30     WiFiDrv::pinMode(BLU,OUTPUT);
31     WiFiDrv::digitalWrite(ORG,LOW);
32     WiFiDrv::digitalWrite(RED,LOW);
33     WiFiDrv::digitalWrite(BLU,LOW);

```

```

1095 void WiFiDrv::digitalWrite(uint8_t pin, uint8_t value)
1096 {
1097     WAIT_FOR_SLAVE_SELECT();
1098     // Send Command
1099     SpiDrv::sendCmd(SET_DIGITAL_WRITE, PARAM_NUMS_2);
1100     SpiDrv::sendParam((uint8_t*)&pin, 1, NO_LAST_PARAM);
1101     SpiDrv::sendParam((uint8_t*)&value, 1, LAST_PARAM);
1102
1103     // pad to multiple of 4
1104     SpiDrv::readChar();
1105
1106     SpiDrv::spiSlaveDeselect();
1107     //Wait the reply elaboration
1108     SpiDrv::waitForSlaveReady();
1109     SpiDrv::spiSlaveSelect();
1110
1111     // Wait for reply
1112     uint8_t _data = 0;
1113     uint8_t _dataLen = 0;
1114     if (!SpiDrv::waitResponseCmd(SET_DIGITAL_WRITE, PARAM_NUMS_1, &_data, &_dataLen))
1115     {
1116         WARN("error waitResponse");
1117         _data = WL_FAILURE;
1118     }
1119     SpiDrv::spiSlaveDeselect();
1120 }

```



Start and Listen on Local Port 8088

```

55 // attempt to connect to WiFi network:
56 while (status != WL_CONNECTED) {
57     Serial1.print("Attempting to connect to SSID: ");
58     Serial1.println(ssid);
59     // Connect to WPA/WPA2 network:
60     status = WiFi.begin(ssid, pass);
61     // wait 10 seconds for connection:
62     delay(10000);
63 }
64 Serial1.println("Connected to WiFi!!");
65 printWifiStatus();
66
67 Serial1.println("\nStarting connection to server...");
68 // if you get a connection, report back via Serial1:
69 Udp.begin(localPort); //unsigned int localPort = 8088;
70 }

```

The screenshot shows the Serial Input/Output Monitor window with the following output:

```

Attempting to connect to SSID: edtpnet2
Connected to WiFi!!
SSID: edtpnet2
IP Address: 192.168.1.75
signal strength (RSSI):-33 dBm

Starting connection to server...

```

The window also displays a hex dump of the received data:

```

0A
53 74 61 72 74 69 6E 67 20 63 6F 6E 6E 65 63 74 69 6F 6E 20 74 6F 20 73 65 72 76 65 72 2E 2E 2E 0D 0A

```

At the bottom, there are input fields for ASCII and HEX, and a status bar showing COM5 BN1 9600, R 0 C 0 R 8 C 1, and a Disconnect button.

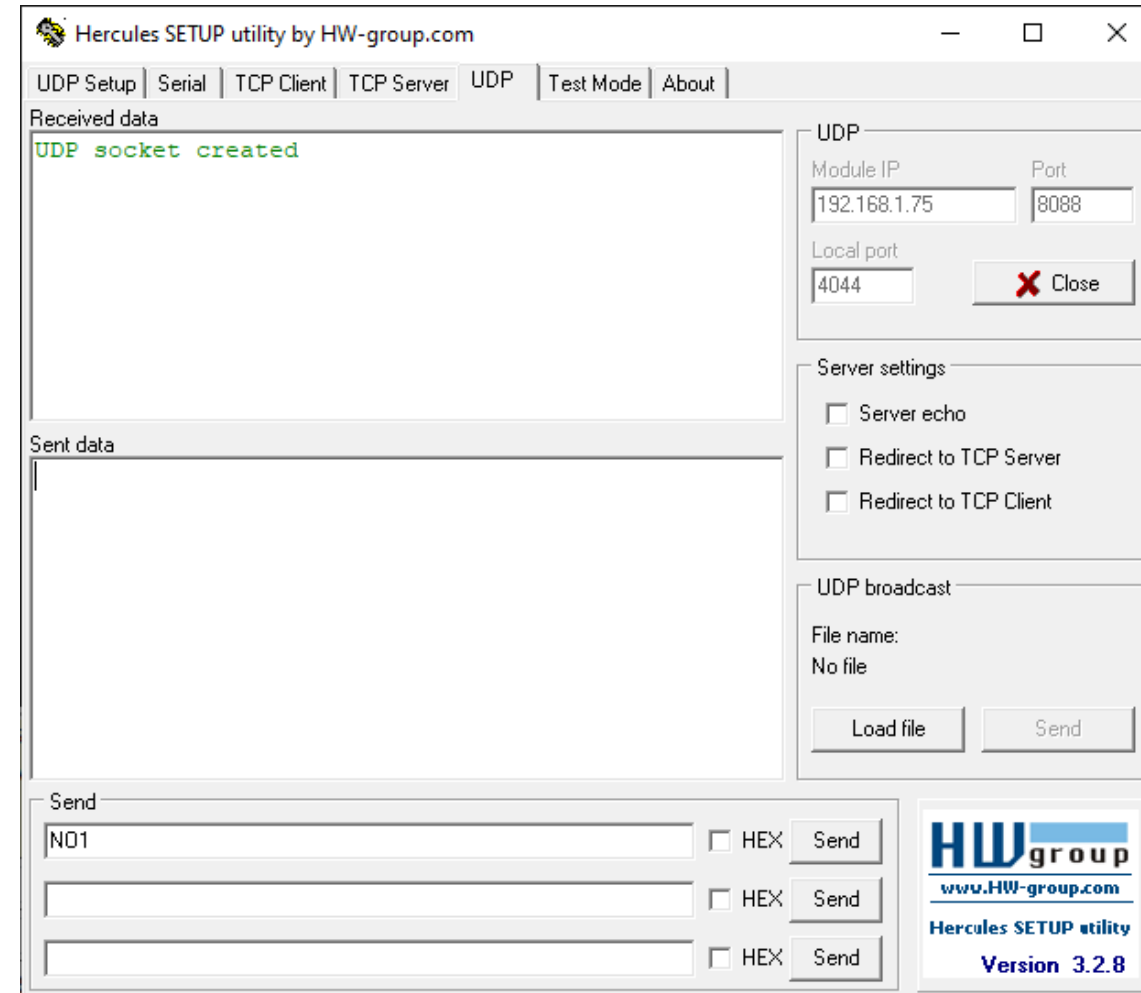
Setup UDP Client/Server

COMMAND FROM PC

```

72 void loop() {
73     sprintf(ackBuffer,"Invalid Command");
74     // if there's data available, read a packet
75     int packetSize = Udp.parsePacket();
76     if (packetSize)
77     {
78         Serial1.print("Received packet of size ");
79         Serial1.println(packetSize);
80         Serial1.print("From ");
81         IPAddress remoteIp = Udp.remoteIP();
82         Serial1.print(remoteIp);
83         Serial1.print(", port ");
84         Serial1.println(Udp.remotePort());
85
86         // read the packet into packetBuffer
87         int len = Udp.read(packetBuffer, 255);
88         if (len > 0)
89         {
90             packetBuffer[len] = 0;
91         }
92         Serial1.println("Contents:");
93         for(i=0;i<packetSize;i++)
94         {
95             Serial1.print(packetBuffer[i],HEX);
96             Serial1.print(" ");
97         }
98         Serial1.println();

```



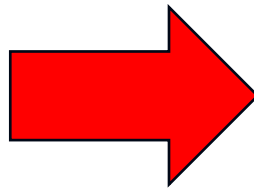
ACKNOWLEDGEMENT FROM RP2040

Parse Received Command to Turn on the ORG LED

```

100 if(packetSize == 3)
101 {
102     switch(packetBuffer[0])
103     {
104         case 0x4E: //"N" nina
105             switch(packetBuffer[1])
106             {
107                 case 0x4F: //"O" ORG
108                     switch(packetBuffer[2])
109                     {
110                         case 0x30: //"0"
111                             WiFiDrv::digitalWrite(ORG,LOW);
112                             sprintf(ackBuffer,"NINA ORG LED = OFF\r\n");
113                             break;
114                         case 0x31: //"1"
115                             WiFiDrv::digitalWrite(ORG,HIGH);
116                             sprintf(ackBuffer,"NINA ORG LED = ON\r\n");
117                             break;
118                     }
119                 break;
120             case 0x52: //"R" RED
121                 switch(packetBuffer[2])
122                 {
123                     case 0x30: //"0"
124                         WiFiDrv::digitalWrite(RED,LOW);
125                         sprintf(ackBuffer,"NINA RED LED = OFF\r\n");
126                         break;
127                     case 0x31: //"1"
128                         WiFiDrv::digitalWrite(RED,HIGH);
129                         sprintf(ackBuffer,"NINA RED LED = ON\r\n");
130                         break;
131                 }
132             break;
133             case 0x42: //"B" BLU
134                 switch(packetBuffer[2])
135                 {
136                     case 0x30: //"0"
137                         WiFiDrv::digitalWrite(BLU,LOW);
138                         sprintf(ackBuffer,"NINA BLU LED = OFF\r\n");
139                         break;
140                     case 0x31: //"1"
141                         WiFiDrv::digitalWrite(BLU,HIGH);
142                         sprintf(ackBuffer,"NINA BLU LED = ON\r\n");
143                         break;
144                 }
145             break;
146         }
147     }

```

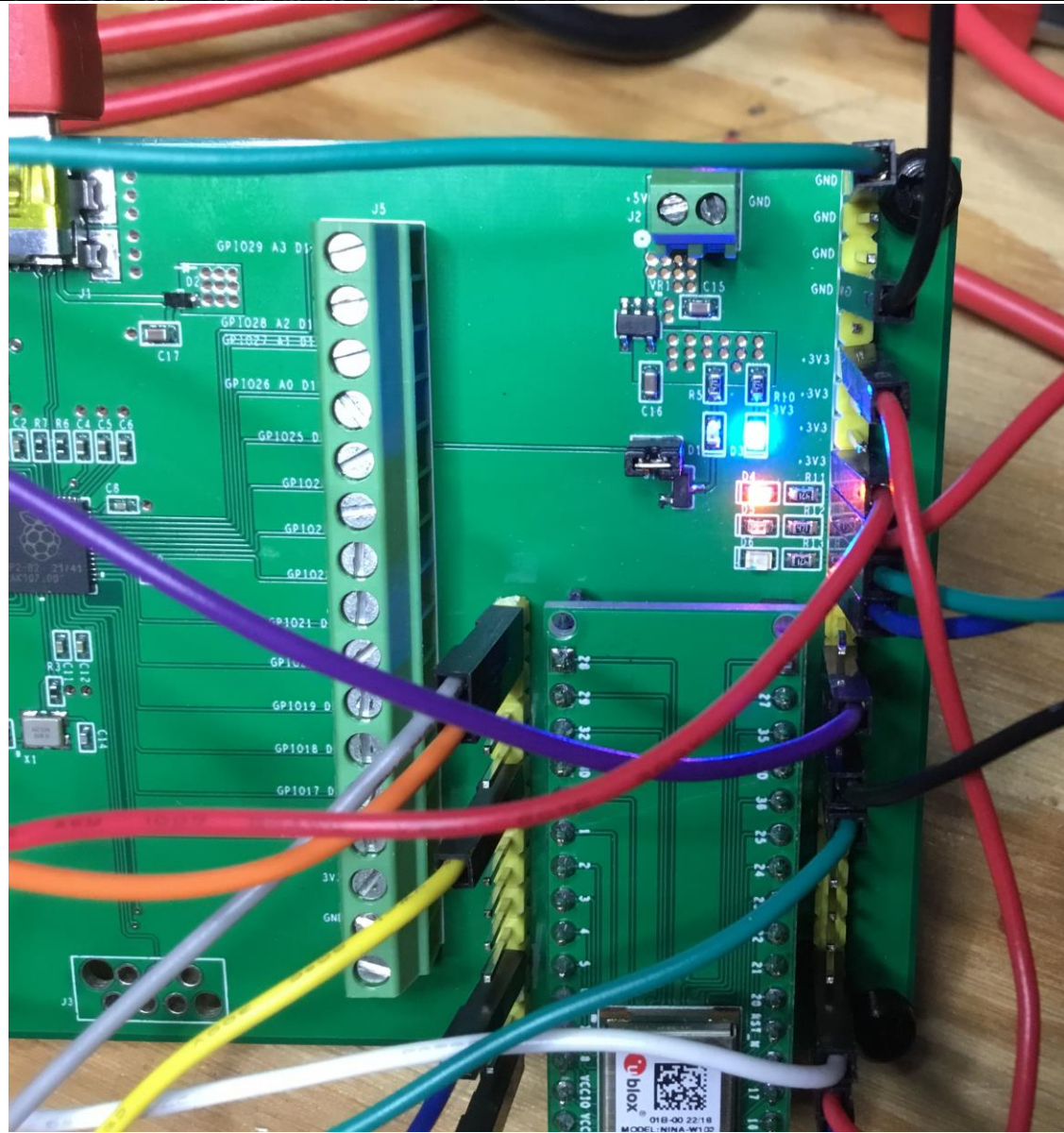


```

100 if(packetSize == 3)
101 {
102     switch(packetBuffer[0])
103     {
104         case 0x4E: //"N" nina
105             switch(packetBuffer[1])
106             {
107                 case 0x4F: //"O" ORG
108                     switch(packetBuffer[2])
109                     {
110                         case 0x30: //"0"
111                             WiFiDrv::digitalWrite(ORG,LOW);
112                             sprintf(ackBuffer,"NINA ORG LED = OFF\r\n");
113                             break;
114                         case 0x31: //"1"
115                             WiFiDrv::digitalWrite(ORG,HIGH);
116                             sprintf(ackBuffer,"NINA ORG LED = ON\r\n");
117                             break;
118                     }
119                 break;

```


Execute the Command and Turn on the ORG LED



Serial Input/Output Monitor

File Edit View Configuration

Input/Output: ASCII, HEX, Line Status, Clear Terminal, Columns, Display, Data Graph, Tools, Ribbon, Classic, Menu Style

```
Attempting to connect to SSID: edtpnet2
Connected to WiFi!!
SSID: edtpnet2
IP Address: 192.168.1.75
signal strength (RSSI):-32 dBm

Starting connection to server...
Received packet of size 3
From 192.168.1.236, port 4044
Contents:
4E 4F 31
```

Hercules SETUP utility by HW-group.com

UDP Setup | Serial | TCP Client | TCP Server | UDP | Test Mode | About

Received data

```
UDP socket created
NINA ORG LED = ON
```

Sent data

```
NO1
```

Server settings

- Server echo
- Redirect to TCP Server
- Redirect to TCP Client

UDP broadcast

File name: No file

Send

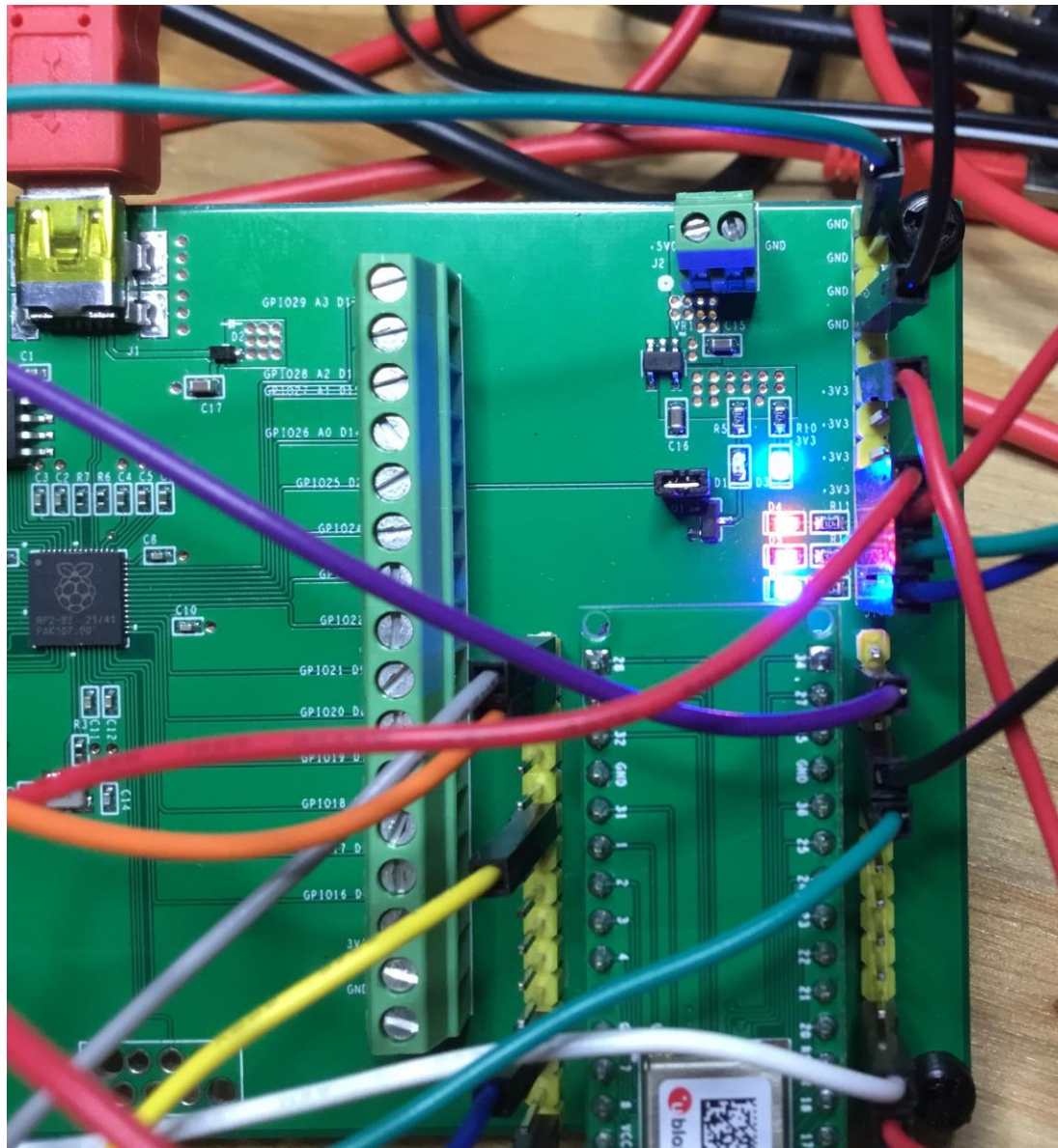
```
NO1
```

DSR DTR DCD RTS CTS RXD Ring TXD Error Break

COM5 8N1 9600 R 0 C 0 R11 C10

HWgroup
www.HW-group.com
Hercules SETUP utility
Version 3.2.8

Execute the Commands to Turn on All NINA LEDs



Serial Input/Output Monitor

Attempting to connect to SSID: edtpnet2
Connected to WiFi!!
SSID: edtpnet2
IP Address: 192.168.1.75
signal strength (RSSI):-32 dBm

Starting connection to server...
Received packet of size 3
From 192.168.1.236, port 4044
Contents:
4E 4F 31
Received packet of size 3
From 192.168.1.236, port 4044
Contents:
4E 52 31
Received packet of size 3
From 192.168.1.236, port 4044
Contents:
4E 42 31
Received packet of size 3
From 192.168.1.236, port 4044
Contents:
4E 4F 30
Received packet of size 3
From 192.168.1.236, port 4044
Contents:
4E 52 30
Received packet of size 3
From 192.168.1.236, port 4044
Contents:
4E 42 30

43 6F 6E 74 65 6E 74 73 3A 0D 0A
34 45 20 34 32 20 33 30 20 0D 0A

ASCII
HEX

DSR DTR DCD RTS CTS RXD Ring TXD Error Break

COM5 8N1 9600 R 0 C 0 R32 C 1 Disconnect

Hercules SETUP utility by HW-group.com

UDP Setup | Serial | TCP Client | TCP Server | UDP | Test Mode | About

Received data
UDP socket created
NINA ORG LED = ON
NINA RED LED = ON
NINA BLU LED = ON
NINA ORG LED = OFF
NINA RED LED = OFF
NINA BLU LED = OFF

Sent data
NO1NR1NB1NO0NRONB0

Send
N00 [] HEX Send
NR0 [] HEX Send
NBO [] HEX Send

UDP
Module IP: 192.168.1.75 Port: 8088
Local port: 4044 [X] Close

Server settings
 Server echo
 Redirect to TCP Server
 Redirect to TCP Client

UDP broadcast
File name: No file
Load file Send

HWgroup
www.HW-group.com
Hercules SETUP utility
Version 3.2.8

Control the "BUILTIN" LED

```

149     case 0x42: //"B" builtin LED
150         switch(packetBuffer[1])
151         {
152             case 0x4F: //"O" ORG LED
153                 switch(packetBuffer[2])
154                 {
155                     case 0x30: //"0"
156                         digitalWrite(LED_BUILTIN,LOW);
157                         sprintf(ackBuffer,"BUILTIN LED = OFF\r\n");
158                         break;
159                     case 0x31: //"1"
160                         digitalWrite(LED_BUILTIN,HIGH);
161                         sprintf(ackBuffer,"BUILTIN LED = ON\r\n");
162                         break;
163                 }
164             break;
165         }
166     break;
167 }
168 }
169 // send a reply, to the IP address and port that sent us the packet we received
170 Udp.beginPacket(Udp.remoteIP(), Udp.remotePort());
171 Udp.write(ackBuffer);
172 Udp.endPacket();
173 }
174 }

```

Hercules SETUP utility by HW-group.com

UDP Setup | Serial | TCP Client | TCP Server | UDP | Test Mode | About

Received data
UDP socket created
BUILTIN LED = ON

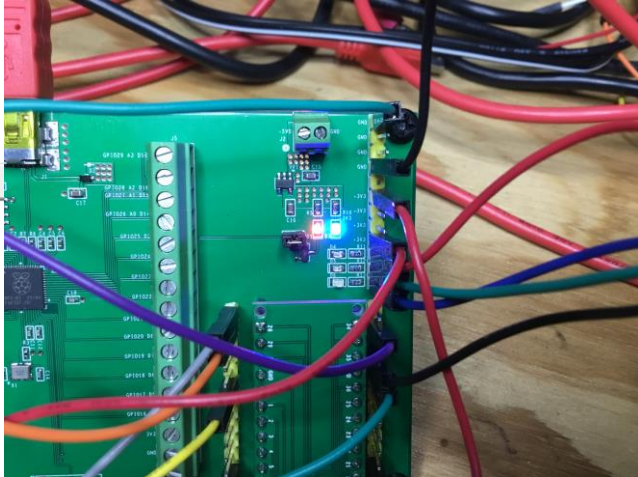
Module IP: 192.168.1.75 | Port: 8088
Local port: 4044

Server settings
 Server echo
 Redirect to TCP Server
 Redirect to TCP Client

UDP broadcast
File name: No file

Send
B00 [] HEX [Send]
B01 [] HEX [Send]
[] HEX [Send]

HWgroup
www.HW-group.com
Hercules SETUP utility
Version 3.2.8

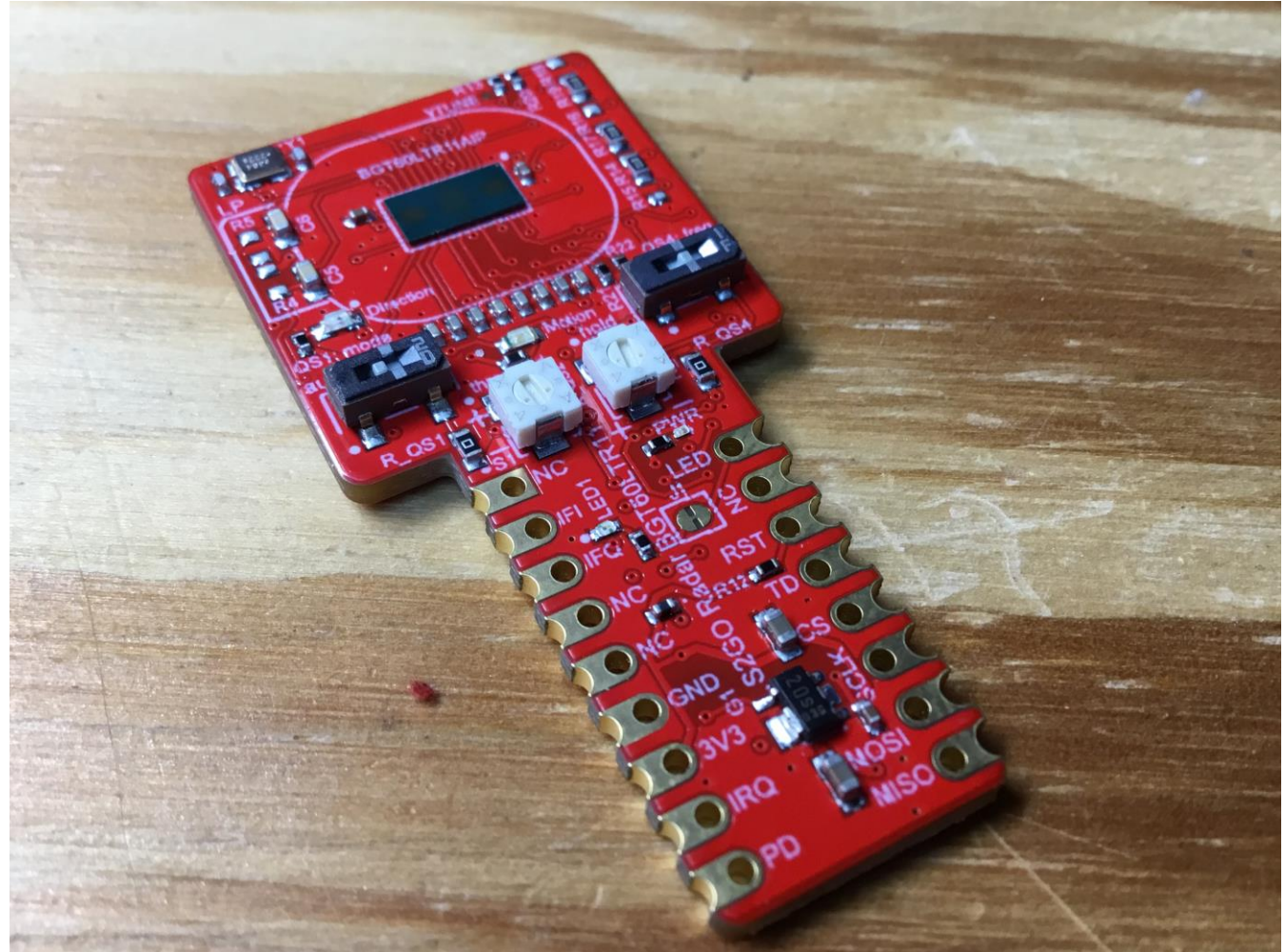


Thank you for attending!!!

Please consider the resources below:

- arduino.cc
- raspberrypi.org
- u-blox.com

MORE TO COME..





DesignNews

Thank You

Sponsored by

DigiKey

