12.Wifi 透传实验例程说明

12.1 开发板简介：

PGX-Nano 开发板搭载了一个 esp32 模块，利用此模块，可使板卡以 WIFI、蓝牙的方式进行通信，应注意的是，若想通过指令对 esp32 模块进行配置从而进入 wifi、蓝牙通信模式，需先对 esp32 模块进行 AT 固件的烧录，具体烧录步骤与烧录所需工程请参考 ESP32 固件烧录指南文档与 esp32_at 文件夹的 FPGA 工程。

使用开发板上的 ESP32 模组进行 WIFI/蓝牙通信时，需要先对模组配置相应的指令，使其进入通信状态，进入通信状态后，开发板即可以开始进行 wifi/蓝牙通信。

12.2 开发板 ESP32 使用简介：

PGX-Nano 搭载一个 esp32 模组，型号为：ESP32-WROVER-IE；此型号模组集成集成 ESP32 系列 ESP32-D0WD-V3 芯片，其两个 CPU 核都可以被单独控制，并且支持 wifi 与蓝牙通信。模组具有丰富的外设接口，开发板选用串口与 esp32 进行数据的交互，ESP32 将交互的数据通过 wifi/蓝牙发送或者接收，从而使板卡进行 wifi/蓝牙通信。

开发板与 ESP32 模组相连的 6 个信号的功能如下：

信号	功能
FPGA_TX	ESP32 的 UART 通信发送接口，用于与开发板的数据交互，ESP32 **指令配置**回复接口
FPGA_RX	ESP32 的 UART 通信接收接口，用于与开发板的数据交互，ESP32 **指令配置**发送接口
BT_INT	Boot 信号，板卡复位结束，或者上电时，会检测此信号的电平，检测到高电平时，进入通信状态，检测到低电平时，进入 AT 固件下载状态。
BOOTH_EN	ESP32 模组使能信号，高电平时 ESP32 模组正常工作，低电平时，ESP32 模组处于复位状态。
BT_TX	ESP32 模组 **AT 固件**下载对应 UART 发送接口。
BT_RX	ESP32 模组 **AT 固件**下载对应 UART 接收接口。

使用开发板上 ESP32 模模组进行 WIFI/蓝牙通信的具体思路如下：

首先：

ESP32 模块在使用前需要先烧录 AT 固件，烧录完毕后，才能正常使用指令配置 ESP32 模块。

烧录 AT 固件时，使用 ESP32 上 BT_TX、BT_RX 两个串口信号进行 AT 指令的烧录。将 ESP32 模组的 BT_TX、BT_RX 信号与板卡的串口信号 FPGA_UART_TX、FPGA_UART_RX 相连，则可以使用板卡串口为 ESP32 模组下载固件。

ESP32 有两种状态，AT 固件下载状态，正常使用状态，以上电时或者复位结束时 BT_INT 信号的电平高低，判断进入那种状态，BT_INT 为高电平时，ESP32 进入正常使用状态，BT_INT 为低电平时，ESP32 进入 AT 固件下载状态。由此可知，在 ESP32 模组上电后，控制使能信号 BOOTH_EN、BT_INT 信号，即可使 ESP32 进入 AT 固件下载状态。

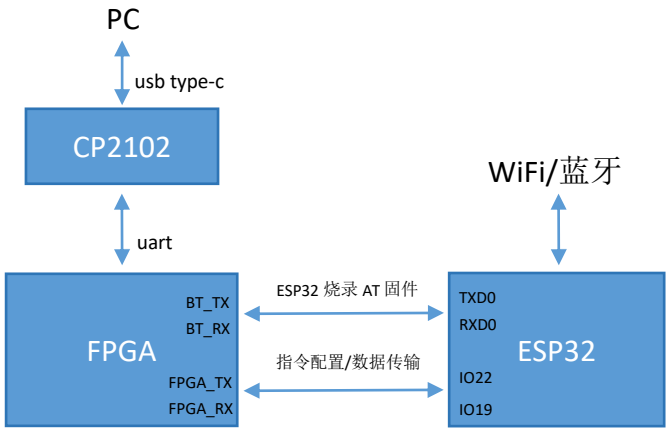
其次：

使用开发板上的 ESP32 模组进行 WIFI/蓝牙通信时，需要先对模组配置相应的指令，使其进入通信状态，进入通信状态后，开发板即可以开始进行 wifi/蓝牙通信。

首先，配置指令时，可以将 ESP32 用于数据通信 UART 信号 FPGA_TX、FPGA_RX 与开发板串口信号 FPGA_UART_TX、FPGA_UART_RX 连接，将板卡串口与 PC 端进行连接，通过 PC 端串口调试助手对 ESP32 模组进行指令的配置。

其次，配置指令后，ESP32 模组就可以完成通过 wifi/蓝牙发送来自与串口的数据以及接收 WIFI/蓝牙的数据通过串口发送给开发板的功能，若希望使用板卡与 ESP32 进行数据的交互，就需要使用判断条件，断开 FPGA_TX、FPGA_RX 与开发板串口信号 FPGA_UART_TX、FPGA_UART_RX 的连接，使用 FPGA 直接向 ESP32 模组的 FPGA_RX 发送数据，或者接收 ESP32 模组的 FPGA_TX 数据，

在 WIFI/蓝牙透传状态下，在结束数据传输前，向 ESP32 模块发送不加换行灯信号的连续+++信号，即可退出透传状态。



12.3 实验目的：

使用板卡实现 UART wifi 透传，使用完毕后拨下拨码开关退出透传状态。

12.4 实验要求：

打开 ssom 文件夹下 sscom5.13.1.exe，该串口调试助手可作为 TCP 测试工具，将其设置为 TCP 服务端，将板卡串口与 PC 端相连，在 PC 端新打开一个串口调试助手为板卡 esp32 配置指令，根据实验步骤使 esp32 进入 wifi 透传状态后，拨动拨码开关 SW0 开启板卡的 wifi 透传通信，按动按键 S2，板卡会通过 WiFi 向 TCP 服务端发送一串 www.meyesemi.com 的字符，并且使用 TCP 服务端发送 0~9 的字符型数据（取消勾选加回车换行），将会显示在板卡数码管上。

12.5 实验原理：

透传原理：

透传即数据传输过程中，发送方发送数据的内容和长度与接收方接受到数据的内容和长度完全相同，不对数据进行任何处理。

实验中，板卡上的 esp32 模组在进入透传状态后，作为数据的发送方，将模块串口接收的数据，直接转化为无线 TCPIP 协议包的数据内容发送出去，此过程即为透传。

12.6 实验源码

顶层源码如下列所示（完整源码请前往实验例程查看）：

由于在使用 ESP32 模组在进行 WIFI/蓝牙透传前，需要先进行指令的配置，因此设计一个拨动开关作为判断条件，在拨动开关为低电平时，板卡串口与 ESP32 模组数据传输对应串口相连，此时 PC 端串口调试助手输出的指令将同过板卡串口进入 ESP32 模组串口，从而实现指令的配置，在拨动开关为高电平时，由 FPGA 直接接管 ESP32 模组数据传输对应串口，按下按键 S2，板卡将发送一串字符，并且将 ESP32 模组数据传输串口传出的数据显示在数码管上。

传输结束，退出透传状态时，只需拨下拨动开关，根据程序设计，FPGA 会自动发送“+++”以结束透传状态。

```
module wifi_test(
   input  wire    clk ,
   input  wire    rst_n ,
   input  wire    key ,
   input  wire    ctrl ,
   input  wire    fpga_uart_rx ,
   input  wire    fpga_tx ,
   output reg     fpga_rx ,
   output reg     fpga_uart_tx ,
   output wire    en ,
   output wire    boot ,
   output wire [7:0]sel ,
   output wire [15:0]seg
```

```
   );

wire  tx ;
wire  [7:0]tx_data ;
wire  tx_pluse ;
wire  [7:0]rx_data ;
wire  rx_en ;
wire tx_busy ;
wire trsn_end ;
wire rx_finish ;
wire tx_end ;
wire seg_flag ;
wire [1:0]ctrl_flag ;

assign en = 1'b1 ;
assign boot = 1'b1 ;

wire [1:0]key_flag ;
assign key_flag[0] = ctrl ;
assign key_flag[1] = key ;
//按键消抖，对一个拨动开关、一个按键进行消抖处理
btn_deb#(
   .BTN_WIDTH (4'd2),
   .BTN_DELAY (26'hF423F)
)btn_deb
(
   .clk          (clk       ),
   .btn_in       (key_flag      ),
   .btn_deb      (ctrl_flag )
);
//产生发送给 ESP32 数据传输对应串口的数据，退出透传时，需要发送“+++”，
此模块检测拨动开关下降沿，检测到后自动发送“+++”结束透传状态
uart_data_gen uart_data_gen(
   .clk                     (clk      ) ,
   .read_data               (rx_data  ) ,
   .ctrl_flag               (ctrl_flag) ,
   .tx_busy                 (tx_busy  ) ,
   .write_max_num           (8'h14    ) ,
   .write_data              (tx_data  ) ,
   .write_en                (tx_pluse ) ,
   .tran_end                (tran_end )
);
//FPGA 产生发送给 ESP32 数据传输对应串口的 RX 端数据
uart_tx #(
```

```
   .BPS_NUM  (16'd434)
)uart_tx
(
   .clk                        (clk     ) ,
   .tx_data                    (tx_data ) ,
   .tx_pluse                   (tx_pluse) ,
   .tx_end                     (tx_end  ) ,
   .uart_tx                    (tx      ) ,
   .tx_busy                    (tx_busy )
);
```

//FPGA 接收来自于 ESP32 数据传输对应串口 TX 端数据

```
uart_rx # (
   .BPS_NUM  (16'd434)
)uart_rx
(
   .clk                       (clk     ) ,

   .uart_rx                   (fpga_tx ) ,

   .rx_data                   (rx_data   ) ,

   .rx_en                     (rx_en       ) ,

   .rx_finish                 (rx_finish       )
);
```

```
reg [1:0]state ;
parameter idle = 2'd0 ;
parameter state_send = 2'd1 ;
parameter state_end = 2'd2 ;
```

//状态机判断波动开关状态，波动开关为关时，板卡串口与 ESP32 模组数据传输串口相连，高电平时，由 FPGA 直接控制 ESP32 模组数据传输串口

```
always @(posedge clk ) begin
   case (state)
      idle : begin
         if (ctrl_flag[0])
            state <= state_send ;
         else
            state <= idle ;
      end
      state_send : begin
         if (tran_end)
            state <= state_end ;
```

```
         else
            state <= state_send ;
      end
      state_end : begin
         if (tx_end)
            state <= idle ;
         else
            state <= state_end ;
      end
      default: state <= idle ;
   endcase
end
```

```
always @(posedge clk ) begin
      if (state != idle )
         fpga_rx <= tx ;
      else
         fpga_rx <= fpga_uart_rx ;
end
```

```
always @(posedge clk ) begin
   if (!seg_flag)
   fpga_uart_tx <= fpga_tx ;
   else
   fpga_uart_tx <=1'b1 ;
end
```

```
assign seg_flag = (state != idle )? 1'b1 : 1'b0 ;
```

//在拨动开关高电平时，将从 ESP32 模组 TX 端接受到的 0~9 字符型数据显示在数码管上

```
seg_dynamic seg_dynamic(
   .clk                    (clk ) ,
   .rst_n                  (rst_n) ,
   .rx_data                (rx_data ) ,
   .rx_en                  (rx_en ) ,
   .seg_flag               (seg_flag ) ,
   .dig                    (sel ) ,
   .seg                    (seg)
);

Endmodule
```

12.7 实验流程：

1、第一次使用，请烧录 AT 固件（esp32 的 AT 固件烧录的位置为 ESP 模块的 flash 中，因此只需烧录一次即可），详情请参考《esp32 固件烧录指南.pdf》文档。

2、搭建 TCP 服务端：打开 tools 文件夹下 sscom 文件夹，点击 sscom5.13.1.exe 使用此串口调试助手作为 TCP 测试工具，在端口号位置选择：TCPServer，（TCP 测试工具所在 PC 端应与板卡 esp32 模组连接同一个路由），设置端口号为 8080（可选择其他端口号，但建议为 8000 以上），点击开始侦听。

3、连接板卡串口，烧录程序后，使用串口对 esp32 进行指令的配置，此时 sw0 应为未拨动状态。（串口调试助手设置波特率应为 115200）

4、指令配置完成后，拨动拨动开关 sw0 开启板卡的 wifi 透传通信，按动按键 S2，板卡会通过 WiFi 向 TCP 服务端发送一串 www.meyesemi.com 的字符，并且使用 TCP 服务端发送 0~9 的字符型数据（取消勾选加回车换行），将会显示在板卡数码管上。

6、实验结束，拨下拨码开关 sw0，退出 wifi 透传状态，输入指定指令，关闭 TCP 连接。

12.7.1 详情参考下列叙述：

搭建 TCP 服务端：

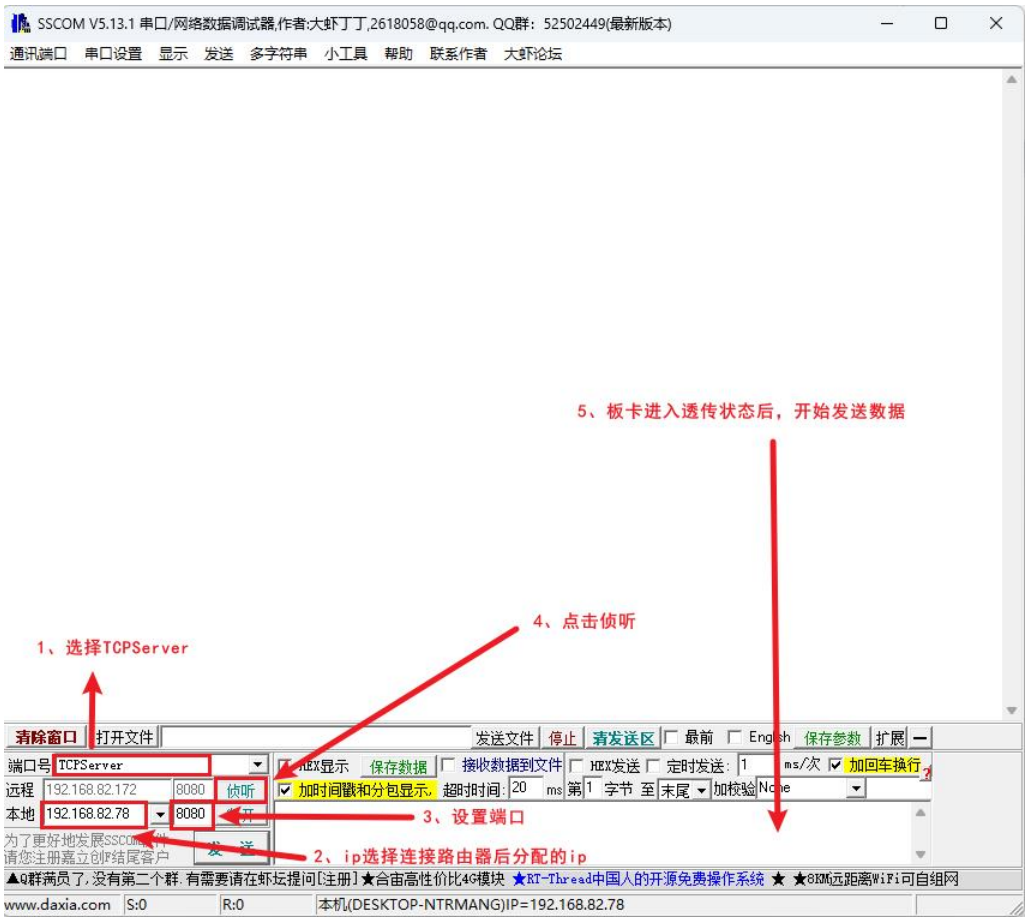
(1) 选择端口号 ： TCPServer

(2) 选择 IP ： IP 应为 PC 连接您路由器后分配的 IP

(3) 设置端口号 ： 8080（可自行设置）

(4) 点击侦听

(5) 等待板卡进行透传状态，板卡进入透传状态后，根据实验流程，发送或接收数据。使用 TCP 服务端发送数据时，请取消勾选“加回车换行”选项



进行 AT 指令配置,进入 wifi 透传状态：

串口调试助手需要勾选加回车换行。

指令发送时前后不能有空格。

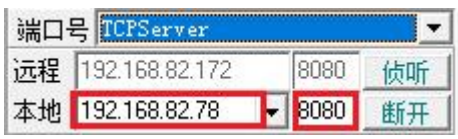
使用串口调试助手发送下列指令：

(1) 指令: AT+CWMODE=1 // Station mode
回复: OK

(2) 指令： AT+CWJAP="WIFI 名称","WiFi 密码" //连接网络，wifi 名称为您路由器的名称，wifi 密码为您路由器的密码
回复： OK

(3) 指令： AT+CIFSR //查看板卡 esp32 模组 IP
回复： OK

(4) 指令： AT+CIPSTART="TCP","192.168.82.78",8080 //ip 地址为 TCP 测试工具的本地 IP，端口号为 TCP 测试工具的本地端口号（端口号可自行设置）



回复： OK

(5) 指令： AT+CIPMODE=1 //进入 WiFi 透传接收模式

回复： OK

(6) 指令： AT+CIPSEND //进入 WiFi 透传模式

回复： OK

实验结束，退出 wifi 透传状态，关闭 TCP 连接：

(1) 将串口调试助手取消勾选加回车换行，发送+++（注意不要有空格）。

(2) 重新勾选串口调试助手的加回车换行选项,发送下列指令。

(3) 指令： AT+CIPMODE=0 //退出 WIFI 透传模式

回复： OK

(4) 指令： AT+CIPCLOSE //退出 TCP 连接

回复： OK

完成 AT 指令配置过程如下：

```
AT+CWMODE=1

OK
AT+CWJAP="123","1234567890"
WIFI DISCONNECT
WIFI CONNECTED
WIFI GOT IP

OK
AT+CIFSR
+CIFSR:STAIP,"192.168.82.172"
+CIFSR:STAMAC,"08:d1:f9:ec:8e:60"

OK
AT+CIPSTART="TCP","192.168.82.78",8080
CONNECT

OK
AT+CIPMODE=1

OK
AT+CIPSEND

OK

>12345678
AT+CIPMODE=0

OK
AT+CIPCLOSE
CLOSED

OK
```

12.8 实验现象

指令配置完成后，拨动拨动开关 sw0 开启板卡的 wifi 透传通信，按动按键 S2，板卡会通过 WiFi 向 TCP 服务端发送一串 www.meyesemi.com 的字符，并且使用 TCP 服务端发送 0~9 的字符型数据（取消勾选加回车换行），将会显示在板卡数码管上。

提示：

1、若 esp32 连接不上 TCP 服务器，可以尝试关闭防火墙，或添加防火墙规则以进行解决。

2、更多 at 指令，可前往 esp32 模块厂商乐鑫科技官网获取。