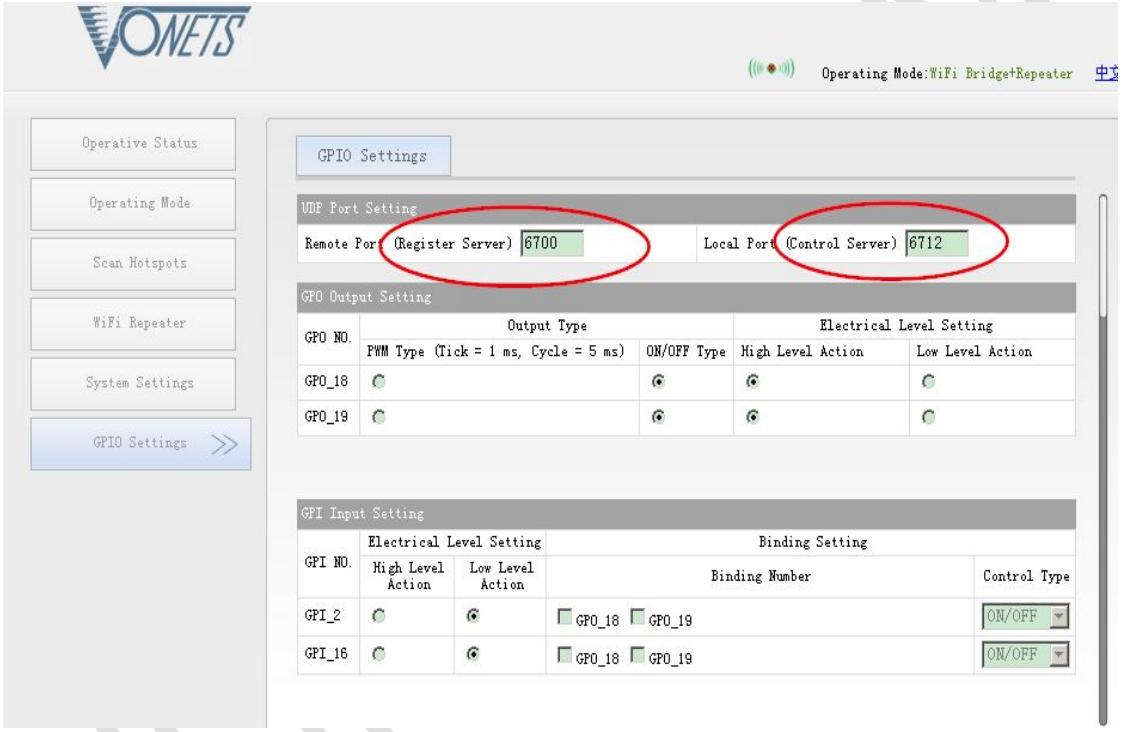
VHA300 GPIO function----APP programming guide

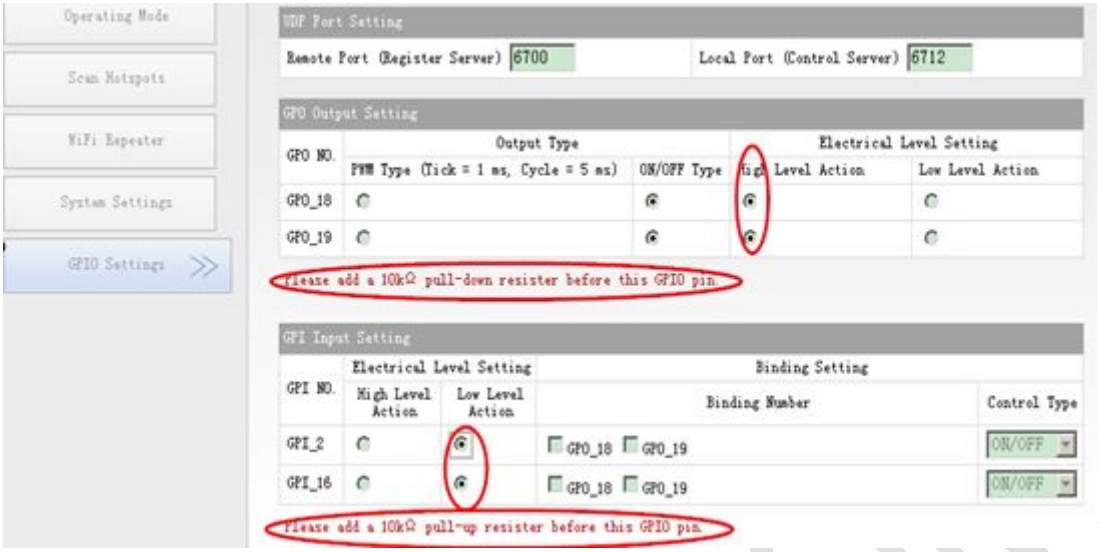
一、Configuration instructions

1、VHA300 using TCP/IP UDP socket communication mode to send GPIO register data(Report the GPIO state changes, generally used for buttons action) and receive GPIO control data(Used for GPIO operation state of the output). Controlling listener port and remote registration service port (APP listener port) can enter the configuration page to manually configure it（http://vonets.cfg. user name：root, password: vonets***pl).

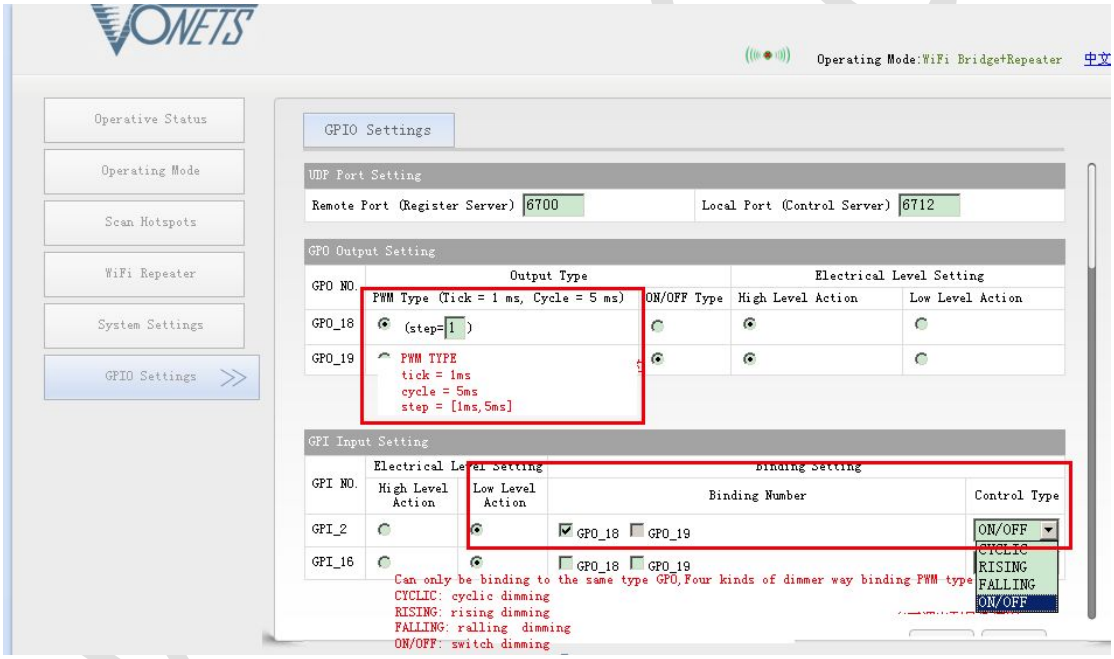


2、According to the configuration page settings, in VHA300 devices configured correctly GPIO pull up and down the resistance:

- Pull up and down resistance is determined by the peripheral circuit design;
- VHA300 specification indicates that there are pulled up and down the resistance of the GPIO port, their pull up and down resistance properties determined by VHA300 module itself;

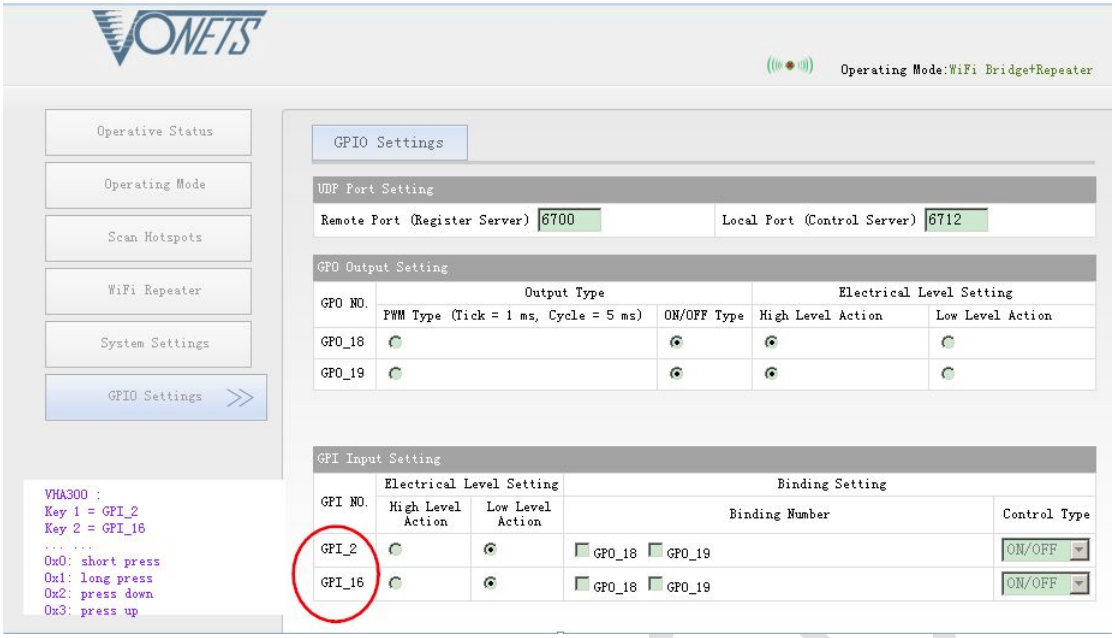


3、According to the demand, configure GPI(IN) and GPO(OUT) working condition.



4、VHA300 will submit four events of GPI to APP listener port:

- 0x00--Short press
- 0x01--Long Press
- 0x02--Press Down
- 0x03--Press Up



二、Outline of the data structure

```
struct vnet_protocol_header     //VNET header data structure
{
    unsigned short protocol;        /*0x1023*/
    unsigned short length;          /*The length of the packet*/
    unsigned char   smac[8];        /*Packet source address,Not in front of the eight 0x0*/
    unsigned char   dmac[8];        /*Packet dest address,Not in front of the eight 0x0*/
    unsigned short service;         /*VNET service type*/
    unsigned short id;              /*Packet ID*/
    unsigned short time;            /*Packet time*/
    unsigned char   key[16];        /*The 0 key*/
    unsigned short code;            /*VNET code type*/
};
```

service: 0x1a20 :Register service;
0x1a21 :Register service response;
0x1a22 :Control service;
0x1a23 :Control service response;

code: 0x2320 :Register request
0x2321/0x2322 :Register request response ACK/NAK
0x3300 :Control register
0x3301/0x3302 :Control request response ACK/NAK

```
struct vnet_gpio_totals    //VNET register GPIO total structure
```

```
{
    unsigned short packet_flag;   /*0x8803*/
    unsigned char   gpio_ttls;
    unsigned short packet_end;    /*0x0d0a*/
};
```

gpio_ttls : The sum channels of VHA300 equipment available, includes input channel and output channel

```
struct vnet_gpi_register     //VNET registered GPI input structure
{
    unsigned char groupid;          /*0xff*/
    unsigned char chlid;            /*pin number*/
    unsigned char chltype;          /*0x1:input type*/
    unsigned char ipttype;          /*0x1:common buttons*/
    unsigned short iptvallen;       /*val len */
    unsigned char *iptvaldata;      /*val data*/
};
```

chltype : 0x1 input type ;
0x2 adj type;
ipttype: 0x1 common buttons
iptvaldata: 0x0 short press，0x01 long press，0x02 press down，0x03 press up

VNET registered GPO output structure

```
struct vnet_gpo_register_switch     //SWITCH TYPE(ON/OFF TYPE)
{
    unsigned char groupid;
    unsigned char chlid;
    unsigned char chltype;
    unsigned char adjtype;    /*0x01:switch type(ON/OFF type) 0x02:PWM type*/
    unsigned long retval;     /*return val*/
};
```

adjtype: 0x1: switch type(ON/OFF type)
0x2: PWM type(PWM Dimming)

```
struct vnet_gpo_register_pwm     //PWM TYPE
{
    unsigned char groupid;
    unsigned char chlid;
    unsigned char chltype;
    unsigned char adjtype;
    unsigned short tick;        /*tick = 1ms*/
```

```
    unsigned char step;       /*step = 1 tick,default 1tick*/
    unsigned short cycle;     /*cycle= 0x5ms*/
    unsigned long retval;
};
```

adjtype: 0x1: switch type(ON/OFF type)
0x2: PWM type

VNET control data structure, which is used to control the output state of the GPO.

```
struct vnet_gpio_control
{
    unsigned char groupid;/*ff*/
    unsigned char chlid;
    unsigned char chltype; /*0x02*/
    unsigned char adjtype; /*0x01/0x02*/
    unsigned char acttype;/*Types of control:*/
    unsigned short time;
    unsigned short adjvallen;
    unsigned char *adjvaldata;
};
```

chlid : pin number
adjtype: 0x1: switch type(ON/OFF type)
0x2: PWM type
acttype:0x0 (the absolute value)
0x1:relative
adjvallen: switch type = 0x2 , pwm type = 0x3
adjvaldata:
switch type :[0]:0x01:LED state flip，0x00:LED OFF, 0xff:LED ON
[1]: 0x0 GPI short press，0x01:GPI long press
Pwm type: [0]: 0x00:cycle dimming
0x01:rising dimming
0x02:falling dimming
0x03:ON/OFF dimming
[1]: 0x0 GPI short press，0x01 GPI long press
[2]:step

三、Tectonic packet instructions

1. VHA300 will submit two kinds of data to the remote APP registration service port, if not received the response 4s later, it will resend again.

- Initialize the total number of data channels and all channels to initialized registration data (input GPI channels and output GPO channels registering).
- GPI events data：0x0 short press，0x01 long press，0x02 press down，0x03 press up;
- This version of firmware submit data uses UDP broadcast mode；

2. VHA300 native control service receives control data from remote APP：

- If configured with the binding operation, VHA300 will send GPI event messages to the native GPO data.
- VHA300 PWM control type GPO status has the following kinds:
  0x0--OFF，0x1-- Level1 ON，0x2--Level2 ON, 0x3--Level3 ON，0x4--Level4 ON，0x5--ON;
- This version of the firmware to ignore length, unified as a short press event processing:
- This version of the firmware only simple control, no group control and binding functions.
- The remote APP can send control packets via UDP unicast or broadcast mode ..

3.Specific set of packages：

1） Initialize the GPIO status register:

VNET header+（0x8803+channel totals+0x0d0a ）+（0x8804 +all input and output channel status+0x0d0a ）+ 0x0d0a

[vnet header +]

//register GPIO total structure

[0x8803 + gpio_ttls + 0x0d0a +]

[0x8804 +]

// registered GPI input structure

[group id（1bytes）+ channel id（1bytes）+ channel type（1bytes）+ input type（1bytes）+ input val len（2bytes）+ input val data（input val len bytes）+0x0a0a +]

// registered GPO output structure

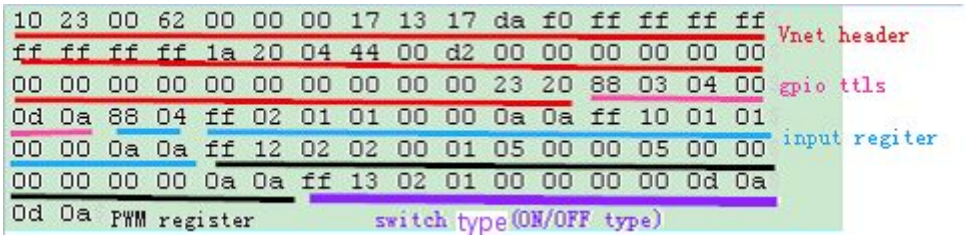
[group id（1bytes）+ channel id（1bytes）+channel type（1bytes）+adj type（1bytes）+tick（2bytes）+step（1bytes）+ pwm cycle（2bytes）+return val（4bytes）+0x0a0a+] (**adj type=0x02/0x03/0x04**)

[group id（1bytes）+channel id（1bytes）+channel type（1bytes）+adj type（1bytes）+ return val（4bytes） + 0x0a0a+] (**adj type=0x01/0x05**)

[0x0d0a]

For example：the following data is VHA300 initialization register data (hexadecimal)

```
10 23 00 62 00 00 00 17 13 17 da f0 ff ff ff ff
ff ff ff ff 1a 20 04 44 00 d2 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 23 20 88 03 04 00
0d 0a 88 04 ff 02 01 01 00 00 0a 0a ff 10 01 01
00 00 0a 0a ff 12 02 02 00 01 05 00 00 05 00 00
00 00 00 00 0a 0a ff 13 02 01 00 00 00 00 0d 0a
0d 0a
```



2） GPI events register data（**input val data=0x0 short press，0x01 long press，0x02 press down，0x03 press up**）。

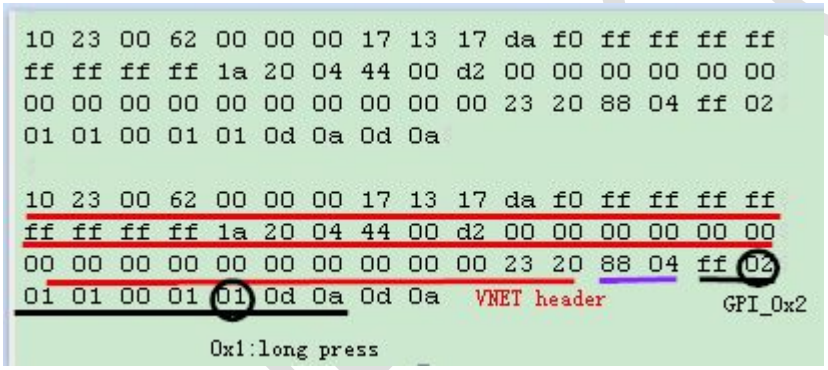
VNET header+（0x8804 +GPI channel status+0x0d0a ）+ 0x0d0a

[vnet header +]

[0x8804 +]

[group id（1bytes）+channel id（1bytes）+ channel type（1bytes）+ input type（1bytes）+input val len（2bytes）+input val data（input val len bytes）+0x0d0a+]

[0x0d0a]



3） Control data

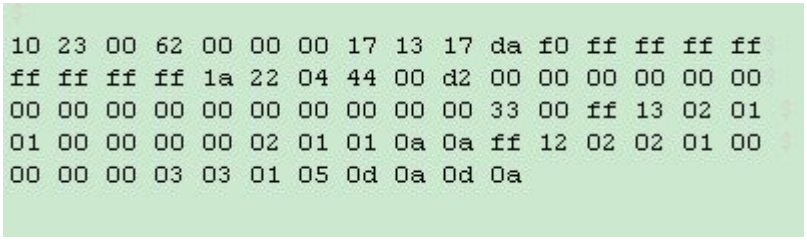
VNET header+（GPO control data+0x0d0a ）+ 0x0d0a

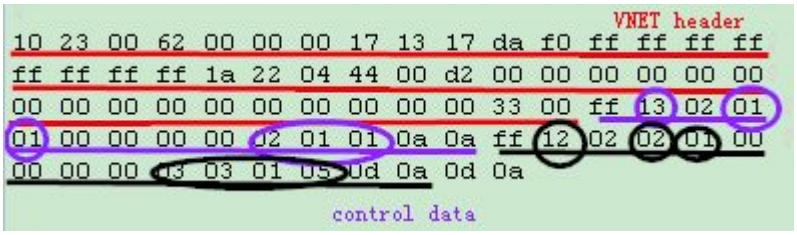
[vnet header +]

[group id（1bytes）+channel id（1bytes）+channel type（1bytes）+adj type（1bytes）+act type(1bytes)

+ live time(2bytes) + adj val len(2 bytes) + adj val data(adj val len bytes) + 0x0a0a/0d0a +]

[0x0d0a]





On Wednesday , November 5, 2014