

Source Codes for DRA818U/DRA818V

V1.01

This document contains source codes for high power wireless voice transceiver module DRA818U/DRA818V and is used only for reference. In order to understand the codes better, users also can refer to the document [ADW1011 configure DRA818V DRA818U through computer](#) for details.

Description

DRA818U/DRA818V modules use standard UART interface to communicate with the host. The default data format is: 9600 8 N 1 (9.6k bps baud rate, 8-bit data, no parity check and 1 stop bit). Users can send AT commands to configure the parameters and the modules will give response when they receive correct AT commands.

AT comannnds:

The AT commands are in ASCII format and ended with <CR><LF>.

Num	AT command	Description	
1	Handshake	Format	AT+DMOCONNECT <CR><LF>
		Response	+DMOCONNECT:0<CR><LF>
2	Group Parameters	Format	AT+DMOSETGROUP=GBW,TFV,RFV,Tx_CTCSS,SQ,Rx_CTCSS<CR><LF>
		Response	+DMOCONNECT: x<CR><LF> x=0:succeded x=1: data out of range
3	Frequency Scan	Format	S+Frequency <CR><LF>
		Response	S=x <CR><LF> x=0: found x=1: no frequency
4	Volume	Format	AT+DMOSETVOLUME=x <CR><LF>
		Response	+ DMOSETVOLUME: X<CR><LF> x=0:succeded x=1: failed
5	Audio filter	Format	AT+SETFILTER=PRE/DE-EMPH,Highpass,Lowpass<CR><LF>
		Response	+ DMOSETFILTER: x<CR><LF> x=0:succeded x=1: failed
6	Tone Tail	Format	AT+SETTAIL=TAIL
		Response	+DMOSETTAIL: X x=0:succeded x=1: failed
7	Read RSSI	Format	RSSI?
		Response	RSSI=X x: RSSI value (0~255)

Table 1: AT Command List

Reference Codes

The codes below are written for microcontroller type PIC1939

RAM DEFINE

Const unsigned char

CMD_HAND[15]={0x41,0x54,0x2B,0x44,0x4D,0x4F,0x43,0x4F,0x4E,0x4E,0x45,0x43,0x54,0x0d,0x0a};

unsigned char CMD_SET[15]={0x41,0x54,0x2b,0x44,0x4d,0x4f,0x53,0x45,0x54,0x47,0x52,0x4f,0x55,0x50,0x3d};

unsigned char

CMD_VOLUME[16]={0x41,0x54,0x2B,0x44,0x4D,0x4F,0x53,0x45,0x54,0x56,0x4f,0x4c,0x55,0x4d,0x45,0x3d};

unsigned char tx_buf[50]={0};

unsigned char rx_buf[30]={0};

unsigned char tx_len;

unsigned char rx_len;

unsigned char len_txnow;

unsigned char len_rxnow;

unsigned char status_cnt = 2;

//=====

void uart_init()

// MCU UART Initialization

{

SPBRGH = 0;

SPBRG = 23;

TXSTA = 0;

RCSTA = 0x90;

BAUDCON = 0;

TXIE = 0;

RCIE = 0;

}

//-----

```
void check_uart()
```

```
// send handshake instruction to module regularly to check the connection status of module
```

```
{
```

```
unsigned char i;
```

```
if(Flag.in_rx == 1)
```

```
{
```

```
    rx_cnt --;
```

```
    if(rx_cnt == 0)
```

```
    {
```

```
        Flag.in_rx = 0;
```

```
        Flag.in_tx = 0;
```

```
        Flag.cn_fail = 1;
```

```
        LED_CTCS = LED_OFF;
```

```
        Flag.poweron = 1;
```

```
        fresh_display();
```

```
        return;
```

```
    }
```

```
}
```

```
if((Flag.in_tx == 1)||((Flag.in_rx == 1))) // No interleave sending instruction, to ensure the module properly receiving  
instruction.
```

```
    return;
```

```
send_hand();
```

```
}
```

```
//-----
```

```
void uart_trans_check(void)
```

```
// check if the transmitting instruction is complete or not
```

```
{
```

```
if((Flag.in_tx == 1)&&(len_txnow > tx_len)&&(TXIF == 1))
```

```
{
```

```
stop_TX();

// Send is finished, close the sending UART function.

Flag.in_tx = 0;

Flag.in_rx = 1;

rx_cnt = 2;

len_txnow = 0;

len_rxnow = 0;

tx_len = 0;

clr_tx_buf();

// Initializes the related registers and flags

start_RX();

// receiving function is available
}

}

//-----

void uart_rcv_ack(void)

{

if(Flag.in_rx == 0)

    return;

if(len_rxnow == rx_len)

{

    Flag.in_rx = 0;

    if((rx_buf[rx_len-3] == 0x30)&&(rx_buf[rx_len-2] == 0x0d)&&(rx_buf[rx_len-1] == 0x0a))

        return;

}

// judge return instruction

{

    status_cnt = 2;

    Flag.cn_fail = 0;

    LED_CTCS = LED_ON;

    fresh_display();
```

```
stop_RX();

if((rx_len == 15)&&(Flag.poweron))
{
    Flag.reset = 1;

    Flag.poweron = 0;
}

return;
}

else if(Flag.cn_fail == 1)
    return;

else
{
    status_cnt -= 1;

    if(status_cnt == 0)
// If the instruction returns null continuously, set the flag bit of module connection to failed
    {
        Flag.cn_fail = 1;

        LED_CTCS = LED_OFF;

        Flag.poweron = 1;
    }
}

}

}

//-----

void send_hand()
{
    unsigned char i;

    for(i=0;i<=14;i++)

        tx_buff[i] = CMD_HAND[i];
```

```
// Load the handshake instruction

rx_len = 15;

tx_len = 15;

// Write handshake instruction to send and receive data bytes

len_txnow = 0;

Flag.in_tx = 1;

clr_rx_buf();

// Clear the receive buffer

start_TX();

// send UART function is available

}

//-----

void send_set()

{

unsigned char i;

for(i=0;i<=14;i++)

    tx_buf[i] = CMD_SET[i];

tx_buf[15] = ASCII(Flag.gbw);

tx_buf[16] = ASCII_comma;

ASCII_TFV();

tx_buf[25] = ASCII_comma;

ASCII_RFV();

tx_buf[34] = ASCII_comma;

tx_buf[35] = ASCII(Tx_ctcs_3);

tx_buf[36] = ASCII(Tx_ctcs_2);

tx_buf[37] = ASCII(Tx_ctcs_1);

tx_buf[38] = ASCII(Tx_ctcs_0);

tx_buf[37] = ASCII_comma;

tx_buf[38] = ASCII(sq);
```

```
tx_buf[39] = ASCII_comma;

tx_buf[40] = ASCII(Rx_ctcs_3);

tx_buf[41] = ASCII(Rx_ctcs_2);

tx_buf[42] = ASCII(Rx_ctcs_1);

tx_buf[43] = ASCII(Rx_ctcs_0);

// Instruction of sending data are ASCII

tx_buf[44] = 0x0d;

tx_buf[45] = 0x0a;

// Send instructions all ends with a carriage return line feed (0X0D,0X0A)

rx_len = 16;

tx_len = 46;

// Write handshake instruction to send and receive data bytes

len_txnow = 0;

Flag.in_tx = 1;

clr_rx_buf();

start_TX();

}

//-----

void send_vol()

{

unsigned char i;

for(i=0;i<=15;i++)

    tx_buf[i] = CMD_VOLUME[i];

// load volume instruction

tx_buf[16] = ASCII(vol);

tx_buf[17] = 0x0d;

tx_buf[18] = 0x0a;

rx_len = 17;

tx_len = 19;
```

// write number of bytes to set the volume level to send and receive data

len_txnow = 0;

Flag.in_tx = 1;

clr_rx_buf();

start_TX();

}

//-----

void clr_tx_buf()

// Clear the send buffer

{

unsigned char i;

for(i=0;i<=39;i++)

tx_buf[i]=0;

}

//-----

void clr_rx_buf()

// Clear the receive buffer

{

unsigned char i;

for(i=0;i<=18;i++)

rx_buf[i] = 0;

}

//-----

void start_TX()

// to make it send UART

{

TXEN = 1;

TXIE = 1;

```
}  
  
//-----  
  
void stop_TX()  
  
// close UARTsending  
  
{  
  
TXEN = 0;  
  
TXIE = 0;  
  
}  
  
//-----  
  
void start_RX()  
  
// to make it receive UART  
  
{  
  
CREN = 1;  
  
RCIE = 1;  
  
}  
  
//-----  
  
void stop_RX()  
  
// close UART receiving  
  
{  
  
CREN = 0;  
  
RCIE = 0;  
  
}  
  
//-----  
  
void interrupt ISR_timer(void)  
  
// interrupt handling  
  
{  
  
unsigned char int_temp;  
  
  
  
if(TXIF)  
  
{
```

```
if(Flag.in_tx == 0)

    stop_TX();

// Not in delivery status, interrupted by mistake

else if(len_txnow <= tx_len)

{

    TXREG = tx_buf[len_txnow];

// update sending data

    len_txnow ++;

}

else

    TXIE = 0;

// send over

}

if(RCIF)

{

    NOP();

    if(Flag.in_rx)

    {

        rx_buf[len_rxnow] = RCREG;

// write the returned dato to receive buffer

        if((len_rxnow++) == (rx_len+1))

            stop_RX();

    }

    else

// Not in receiving state, interrupted by mistake, invalid return values

    {

        stop_RX();

        int_temp = RCREG;

    }

}
```

}

}

<p>Dorji Applied Technologies A division of Dorji Industrial Group Co., Ltd</p> <p>Add.: Xinchenuayuan 2, Dalangnanlu, Longhua, Baoan district, Shenzhen, China 518109 Tel: 0086-755-28156122 Fax.: 0086-755-28156133 Email: sales@dorji.com Web: http://www.dorji.com</p>	<p>Dorji Industrial Group Co., Ltd reserves the right to make corrections, modifications, improvements and other changes to its products and services at any time and to discontinue any product or service without notice. Customers are expected to visit websites for getting newest product information before placing orders.</p> <p>These products are not designed for use in life support appliances, devices or other products where malfunction of these products might result in personal injury. Customers using these products in such applications do so at their own risk and agree to fully indemnify Dorji Industrial Group for any damages resulting from improper use.</p>
---	---