Network communication protocol for SNOWLeoSDR platform

Version	Date	Revision
V1.0	23/12/2013	Initial version
V1.1	26/05/2014	Add SNOWLeoSDR support

1、Control Word Definitions

```
#define  PACKAGE_HEAD              0xF0
#define  DMA_FILE_TX               0x10
#define  DMA_FILE_RX               0x11
#define  DMA_START_ADC             0x12
#define  DMA_START_DAC             0x13
#define  DMA_STOP_ADC              0x14
#define  DMA_STOP_DAC              0x15
#define  DMA_CONNECT_DISABLE       0x16
#define  SDR_RF_CTRL_TX_FREQ       0x17
#define  SDR_RF_CTRL_RX_FREQ       0x18
#define  SDR_RF_CTRL_TX_VGA        0x19
#define  SDR_RF_CTRL_RX_VGA        0x20
#define  SDR_RF_CTRL_TEST_SEQ      0x21
```

2、Network Command Format

Network command length is 8 bytes,command format as follows

MSB LSB

packet head	Control Word	Data	Data	Data	Data	Data	Data

3、Network Command Details

DMA_FILE_TX ----PC will send a file to snowleosdr, the file size is SIZE Kbytes

MSB LSB

F0	10	0	MODE	SIZE(in Kbyte)

MODE：Receive File Mode：”1” Single，”0”Loop default:1

DMA_FILE_RX ----PC will recv a file from snowleosdr, the file size is SIZE Kbytes

MSB LSB

F0	11	0	0	SIZE(in Kbyte)

DMA_START_ADC ----Start ADC channel

MSB LSB

F0	12	0	0	0	0	0	0

DMA_START_DAC ----Start DAC channel

MSB LSB

F0	13	0	0	0	0	0	0

~~**DMA_STOP_ADC ----Close ADC channel**~~

~~**MSB LSB**~~

~~F0~~	~~14~~	~~0~~	~~0~~	~~0~~	~~0~~	~~0~~	~~0~~

~~**DMA_STOP_DAC ----Close DAC channel**~~

~~**MSB LSB**~~

~~F0~~	~~15~~	~~0~~	~~0~~	~~0~~	~~0~~	~~0~~	~~0~~

DMA_CONNECT_DISABLE ----Disconnect pc and snowleosdr’s network connection

MSB LSB

F0	16	0	0	0	0	0	0

SDR_RF_CTRL_TX_FREQ ----Snowleosdr’s TX RFconfiguration

MSB LSB

F0	17	Frequency (MHz)	0	0	0	0

SDR_RF_CTRL_RX_FREQ ---- Snowleosdr’s RX RFconfiguration

MSB LSB

F0	18	Frequency (MHz)	0	0	0	0

SDR_RF_CTRL_TX_VGA ---- Snowleosdr’s TX Gainconfiguration

MSB LSB

F0	19	VGA2	VGA1	PA	GPIOSEL	0	0

~~VGA：8'd0~8'd63,对应0~31.5dB~~

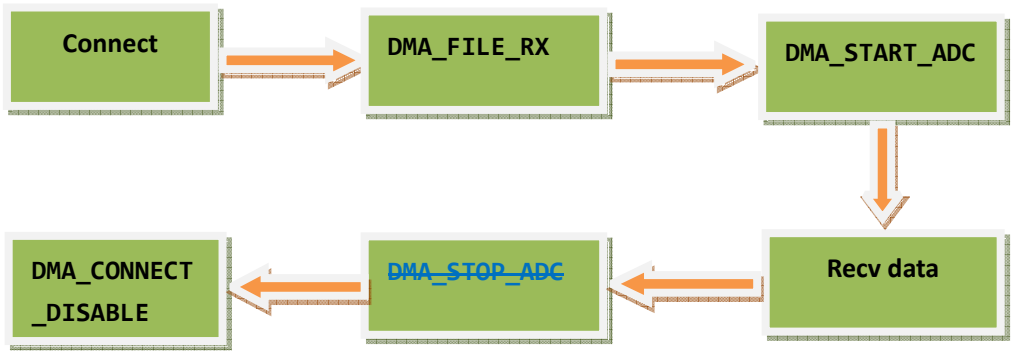
SDR_RF_CTRL_RX_VGA ---- Snowleosdr’s RX Gainconfiguration

MSB LSB

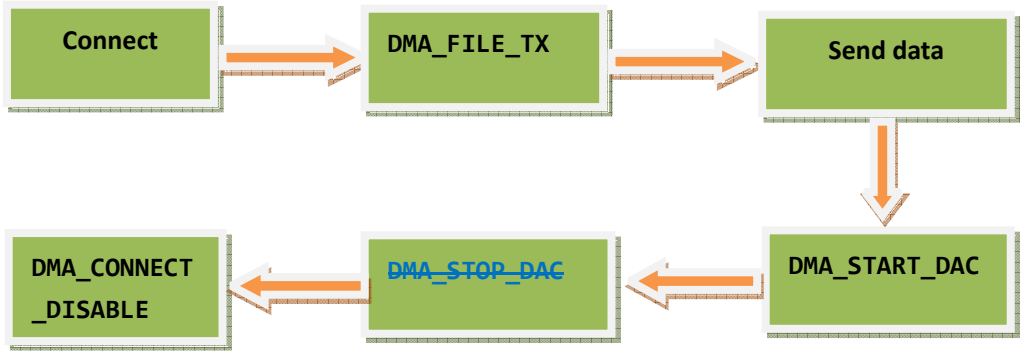
F0	20	LNA	VGA	GPIOSEL	0	0	0

4、Operation Flow

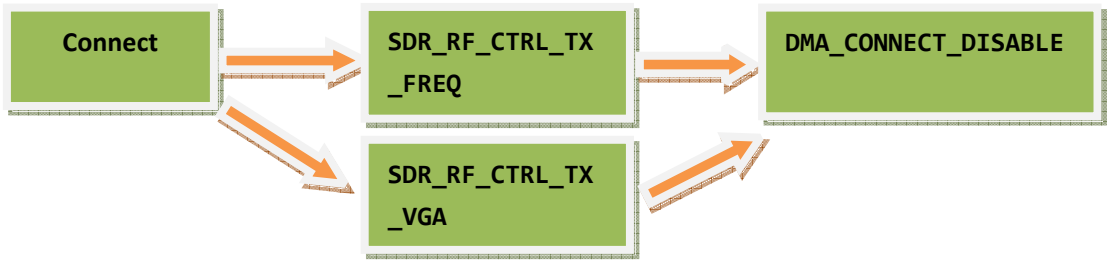
ADC



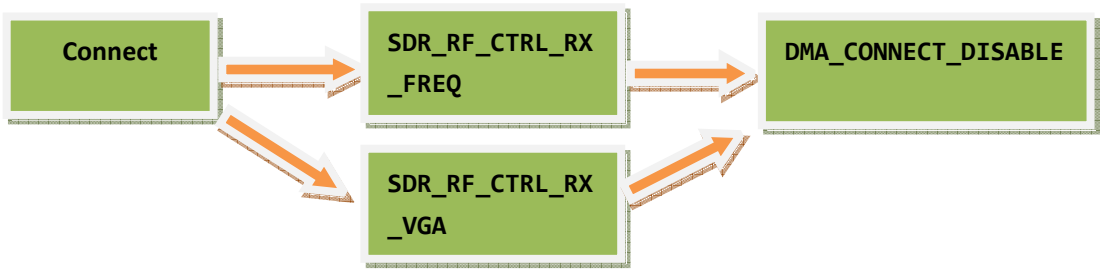
DAC



TX RF Configure



RX RF Configure



Sample Code

Configure TX RF

C code

```
int set_freq(unsigned int d_tx_freq)
{
        unsigned int cmd_buf[2]={0,0};
        d_tx_freq = d_tx_freq & 0x000000FF;
        cmd_buf[0] = 0xF0170000|(d_tx_freq<<8); /*set tx freq*/
        if(send(sockfd, cmd_buf, sizeof(cmd_buf), 0) > 0)
                return 1;
        else
                return -1;
}
```

Matlab code

```
t = tcpip('192.168.1.10', 8000);
set(t,'InputBufferSize',64*1024);
set(t,'OutputBufferSize',16*1024);
fopen(t);

%% send rx freq
test_seq=[0 2 hex2dec('17') hex2dec('f0') 0 0 0 0];
fwrite(t,test_seq,'uint8');
```

Recv data from network

C code

```
#define RX_SAMPLES_NUM    8*1024
void *recv_sample(void *)
{
        unsigned int cmd_buf[2]={0,0};
        int len = 0, i = 0;
        cmd_buf[0] = 0xF0110000; /*set rx size*/
        cmd_buf[1] = 0x00000008; /*recv 8KBytes */
        send(sockfd, cmd_buf, sizeof(cmd_buf), 0);

        cmd_buf[0] = 0xF0120000;    //start adc
        send(sockfd, cmd_buf, sizeof(cmd_buf), 0);
        do {
                len += recv(sockfd, data+len, RX_SAMPLES_NUM-len, 0);
        }while(len != RX_SAMPLES_NUM);

        cmd_buf[0] = 0xF0140000; /*stop adc*/
        send(sockfd, cmd_buf, sizeof(cmd_buf), 0);
return NULL;
}
```

Matlab code

```
%%create tcpip connection
link = tcpip('192.168.1.10', 8000);
```

```
set(link,'InputBufferSize',256*1024);
set(link,'OutputBufferSize',16*1024);
fopen(link);
%% send rx size cmd, recv 8KBytes
rx_size=[0 0 hex2dec('11') hex2dec('f0') 80 0 0];
fwrite(link,rx_size,'uint8');
%% send adc start cmd
adc_start=[0 0 hex2dec('12') hex2dec('f0') 0 0 0 0];
fwrite(link,adc_start,'uint8');
%%recv data
data = fread(link,8*1024,'uint8');Send data to network
```

C code

```
int send_sample(unsigned int d_tx_mode)
{
        unsigned int cmd_buf[2]={0,0};
        int len = 0, i = 0, nbyte = 0;

        cmd_buf[0] = 0xF0100000|(d_tx_mode&0x000000FF); //set tx size
        cmd_buf[1] = TX_SAMPLES_NUM/1024;
        send(sockfd, cmd_buf, sizeof(cmd_buf), 0);
                        usleep(10000);
        send(sockfd, read_buffer, TX_SAMPLES_NUM, 0);

        return NULL;
}
```

Matlab code

```
fid1=fopen('F:\matlab_work\tone_16bit.dat','r');
txdata=fread(fid1,'int16');
%%Data Rearrangement
txd1=(txdata<0)*65536+txdata;
txd2=dec2hex(txd1,4);
txd3=txd2(:,1:2);
txd4=txd2(:,3:4);
txd5=hex2dec(txd3);
txd6=hex2dec(txd4);
txd7=zeros(length(txd6)*2,1);
txd7(1:2:end)=txd6;
```

```
txd7(2:2:end)=txd5;
fclose('all');
t = tcpip('192.168.1.10', 8000);
set(t,'InputBufferSize',16*1024);
set(t,'OutputBufferSize',64*1024);
fopen(t);
%% send file tx cmd，send 16Kbytes data
dac_stop=[0 0 hex2dec('10') hex2dec('f0') 16 0 0 0];
fwrite(t,dac_stop,'uint8');
%% send data.
fwrite(t,txd7,'uint8');
%% send dac start cmd
dac_start=[0 0 hex2dec('13') hex2dec('f0') 0 0 0 0];
fwrite(t,dac_start,'uint8');
```

Note: The above codes are the reference code, part of the C code is Pseudo-code