APPENDIX

APPENDIX A

C++ Code for drive of AD9854

```
#ifndef __DDS_H__
#define __DDS_H__

/************************************************
               AD9854
I/O:
          DATA --FSMC A0-A8;
          ADR  --FSMC A9-A14;
          RESET--FSMC A15;
          UDCLK--FSMC A16;
          WR   --FSMC A17;
          RD   --FSMC A18;
          FDATA--FSMC NBL0;
          OSK  --FSMC NBL1;
************************************************/
#include "sys.h"
#include "stdlib.h"

#define uint  unsigned int
#define uchar unsigned char
#define ulong unsigned long

uchar FreqWord[6];               //6-bit freq word

//**********************SYSCLK and relating variables*************************

#define      CLK_Set            12
const ulong  Freq_mult_ulong  = 1172812;
const double Freq_mult_doulle = 1172812.402961067;

//**************************Modify Hardware*********************************


#define AD9854_RST   PFout(15)
#define AD9854_UDCLK PDout(11)
#define AD9854_WR    PDout(12)
#define AD9854_RD    PDout(13)
//#define AD9854_FDATA   //AD9854 FSK,PSK control
//#define AD9854_OSK     //AD9854 OSK control

//******************************************************
 static void IO_Init(void);
 static void AD9854_WR_Byte(uchar addr,uchar dat);
 extern void AD9854_Init(void);
 static void Freq_convert(long Freq);
 extern void AD9854_SetSine(ulong Freq,uint Shape);

//I/O init

void IO_Init(void)
{
```

```
  GPIO_InitTypeDef  GPIO_InitStructure;

  RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOF|RCC_AHB1Periph_GPIOG|RCC_AHB1Periph_GPIOD
, ENABLE);

  GPIO_InitStructure.GPIO_Pin = (0X3F|0X07<<13); //F0-F5,F13-F15
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;//normal output mode
  GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;//Push-pull output
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;//100M
  GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;//pull down
  GPIO_Init(GPIOF, &GPIO_InitStructure);//initialise GPIOF0-5,13-15

  GPIO_InitStructure.GPIO_Pin = 0x3f;//G0-G5
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;//normal output mode
  GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;//Push-pull output
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;//100M
  GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP ;//pull down
  GPIO_Init(GPIOG, &GPIO_InitStructure);//initialise GPIOG0-G5


  GPIO_InitStructure.GPIO_Pin = (0X07<<11); //D11-13
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;//normal output mode
  GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;//Push-pull output
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;//100M
  GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;//pull down
  GPIO_Init(GPIOD, &GPIO_InitStructure);//initialise GPIOD11-13

}
//==============================================================================
//Name      :void AD9854_WR_Byte(uchar addr,uchar dat)
//Function  :AD9854 parellel port write
//Input     :addr 6bit address
//           dat
//Output    :na
//==============================================================================
void AD9854_WR_Byte(uchar addr,uchar dat)
{
    u16 DATA=dat&0x3f;
    u16 data=(dat>>6)&0x03;
    u16 ADDR=addr&0x3f;

    GPIOF->ODR&=~(0X3F|(0x03<<13));
    GPIOF->ODR|=DATA|(data<<13);
    GPIO_Write(GPIOF,DATA|(data<<13));

    GPIOG->ODR&=~(0X3F);
    GPIOG->ODR|=ADDR;
    GPIO_Write(GPIOG,ADDR);


    AD9854_WR=0;
    AD9854_WR=1;
}
//==============================================================================
//Name      :void AD9854_Init(void)
//Function  :AD9854 config
//INPUT:na
//OUTPUT:na
//==============================================================================
void AD9854_Init(void)
```

```
{
    IO_Init();

    AD9854_WR=1;//disable read write
    AD9854_RD=1;
    AD9854_UDCLK=0;
    AD9854_RST=1;                  //reset AD9854
    AD9854_RST=0;

    AD9854_WR_Byte(0x1d,0x00);     //close Comparators
    AD9854_WR_Byte(0x1e,CLK_Set); //set CLK_SET
    AD9854_WR_Byte(0x1f,0x00);     //set system mode 0 update through external
    AD9854_WR_Byte(0x20,0x60);     //set adjustable amplitude
                                   //cancel interpolation compensation
    AD9854_UDCLK=1;                //update AD9854 output
    AD9854_UDCLK=0;
}
//============================================================================
//Name      :void Freq_convert(long Freq)
//Function  :Signal frequency data conversion
//Input     :Freq   Frequency to be converted, 0~SYSCLK/2
//Output    :NA   but effects value for FreqWord[6]
//Note      FTW = (Desired Output Frequency × 2N)/SYSCLK
//          N=48,Desired Output Frequency,Freq,SYSCLK
//          FTW is 48Bit FreqWord[6]
//============================================================================

void Freq_convert(long Freq)
{
    ulong FreqBuf;
    ulong Temp=Freq_mult_ulong;

    uchar Array_Freq[4];     //Split the input frequency factor into four bytes
    Array_Freq[0]=(uchar)Freq;
    Array_Freq[1]=(uchar)(Freq>>8);
    Array_Freq[2]=(uchar)(Freq>>16);
    Array_Freq[3]=(uchar)(Freq>>24);

    FreqBuf=Temp*Array_Freq[0];
    FreqWord[0]=FreqBuf;
    FreqBuf>>=8;

    FreqBuf+=(Temp*Array_Freq[1]);
    FreqWord[1]=FreqBuf;
    FreqBuf>>=8;

    FreqBuf+=(Temp*Array_Freq[2]);
    FreqWord[2]=FreqBuf;
    FreqBuf>>=8;

    FreqBuf+=(Temp*Array_Freq[3]);
    FreqWord[3]=FreqBuf;
    FreqBuf>>=8;

    FreqWord[4]=FreqBuf;
    FreqWord[5]=FreqBuf>>8;
}
//============================================================================

//Name      :void AD9854_SetSine(ulong Freq,uint Shape)
//Function  :AD9854 Sine generator
```

```
//Input     :Freq   0~(1/2)*SYSCLK
//           Shape  12 Bit, (0~4095),the larger selected value, the greater amplitude
//Output    :na
//================================================================================

void AD9854_SetSine(ulong Freq,uint Shape)
{
    uchar count;
    uchar Adress;

    Adress = 0x04;          //Select the initial value of the frequency control address


    Freq_convert(Freq);                 //Freq convert

    for(count=6;count>0;)               //6 bit
    {
        AD9854_WR_Byte(Adress++,FreqWord[--count]);
    }

    AD9854_WR_Byte(0x21,Shape>>8);      //Set I channel amplitude
    AD9854_WR_Byte(0x22,(uchar)(Shape&0xff));

    AD9854_WR_Byte(0x23,Shape>>8);      //Set Q channel amplitude
    AD9854_WR_Byte(0x24,(uchar)(Shape&0xff));

    AD9854_UDCLK=1;                     //Update AD9854 Output
    AD9854_UDCLK=0;
}

#endif
```

APPENDIX B

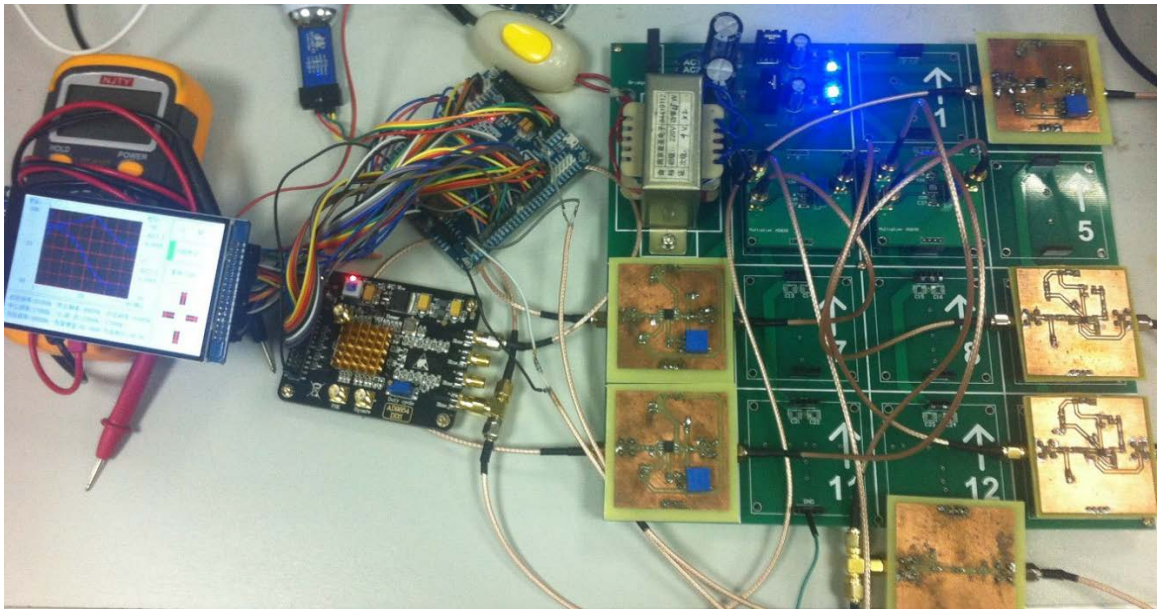

FIGURE 1. FREQUENCY CHARACTERISTICS TESTER USING AD9854 AND STM32 MICROCONTROLLER (ELECFANS, 2014)