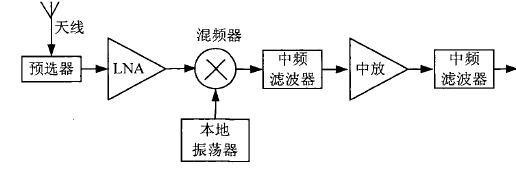
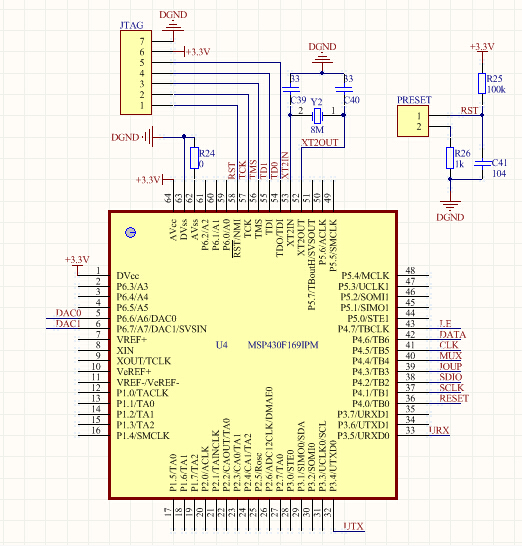
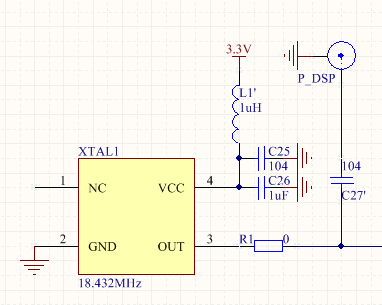
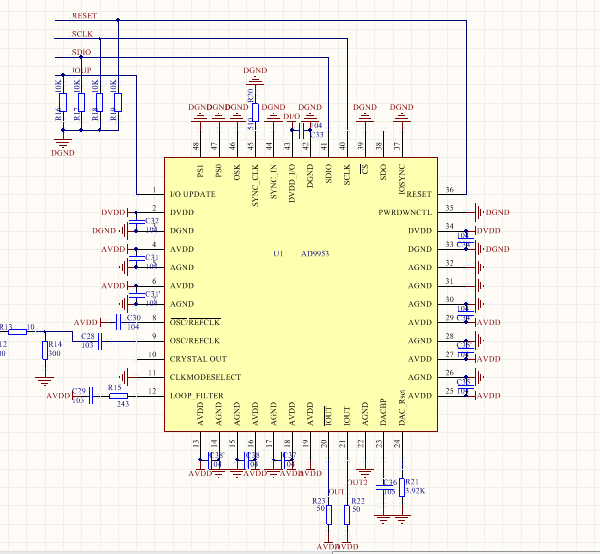
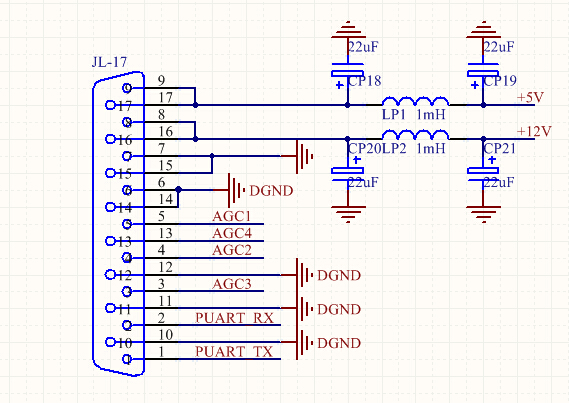
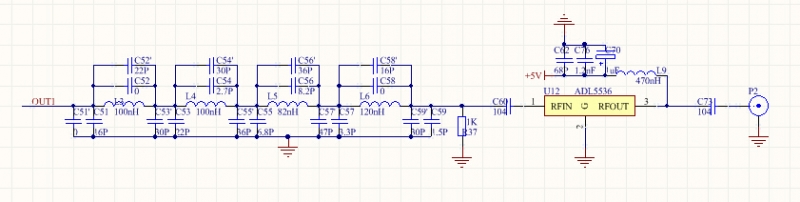
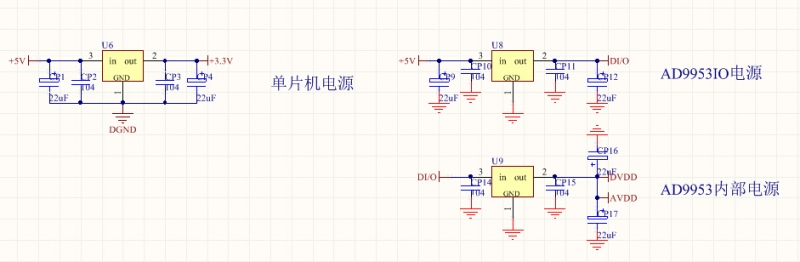
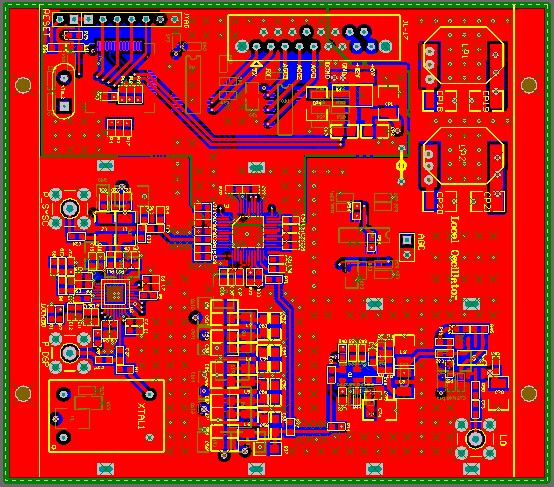
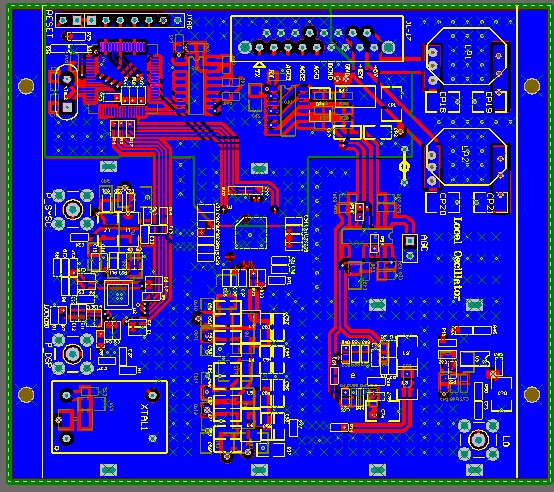
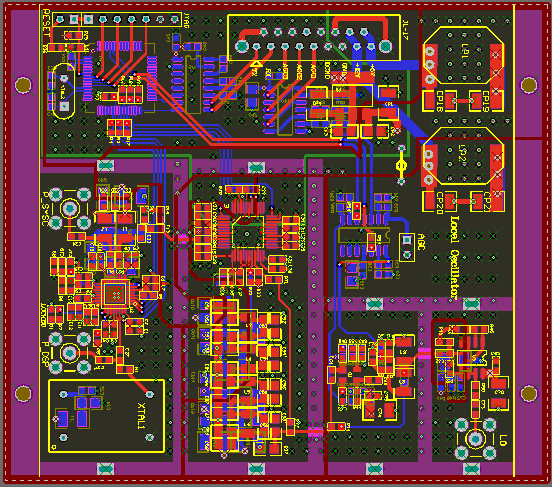
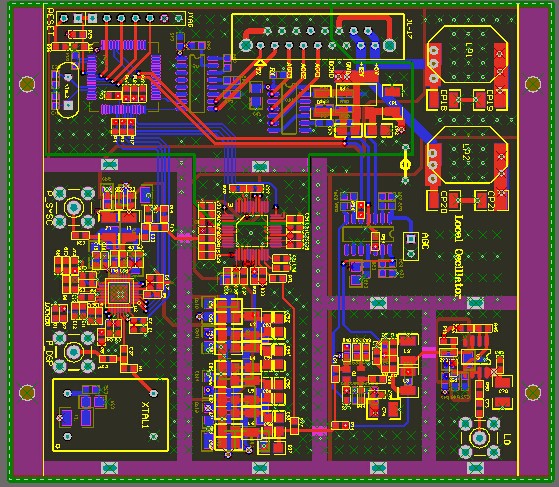
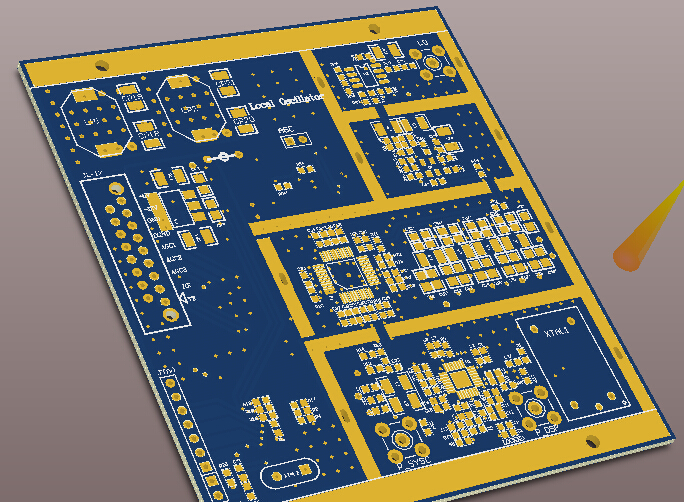
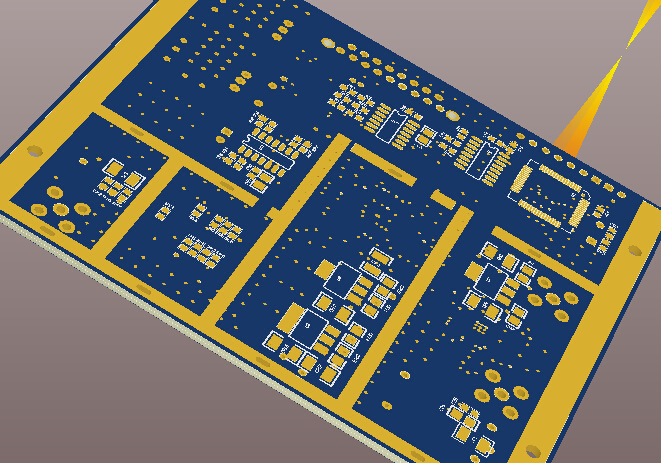
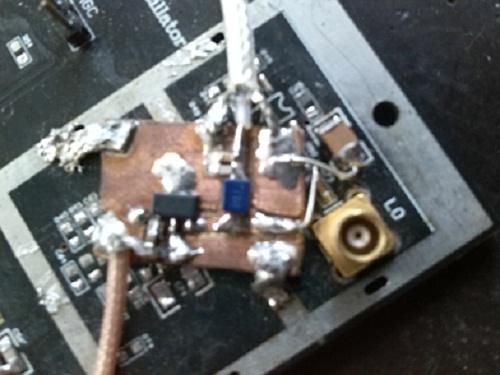
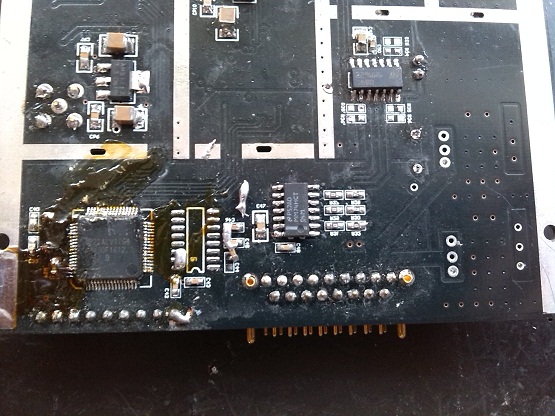
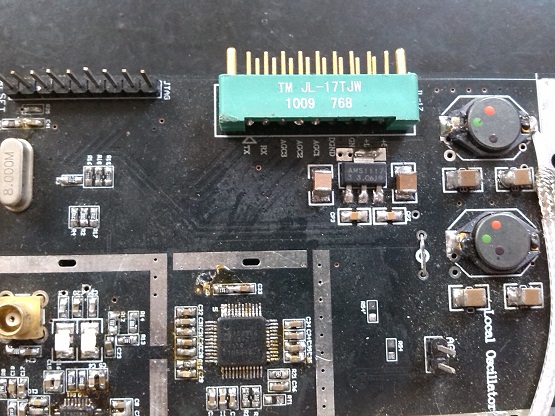
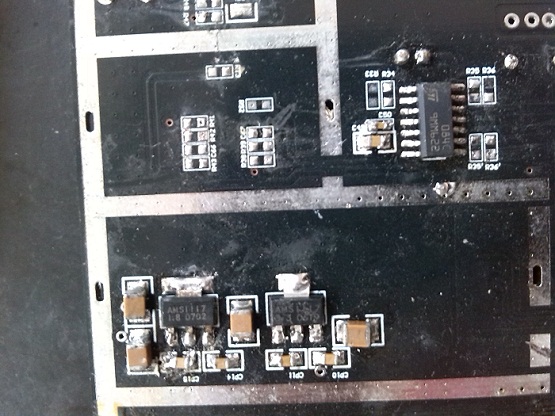
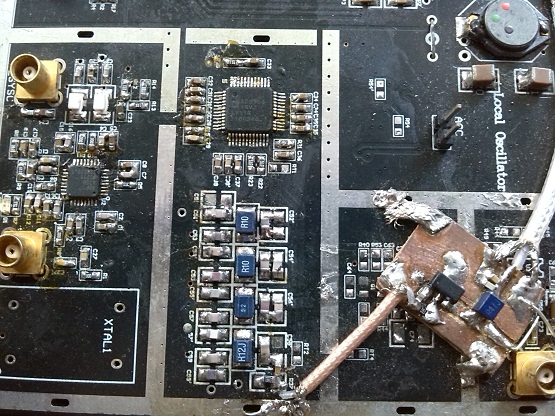
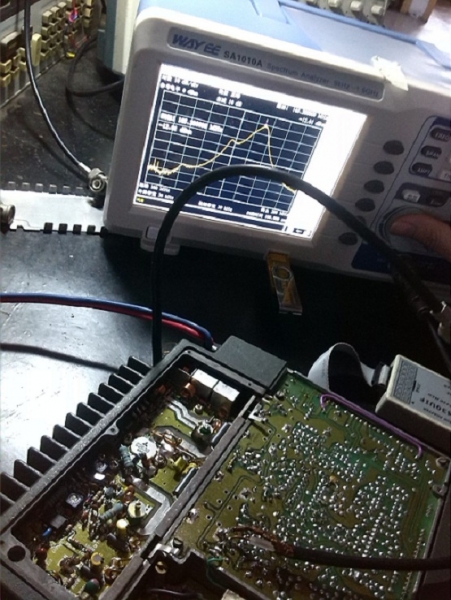
**【电路设计】基于MSP430F169的超外差接收机本振源设计**

超外差式接收机是使用最为广泛一种技术,也是最古老的一种技术。  
目前的接收机大致的原理几乎都是基于这个结构来接受的。  
大致的原理框图如下：  
  
超外差接收机基本原理框图  
  
  
超外差接收机结构就是把高频输入信号与本振信号进行相互混频,后经中频滤波器选出两者的和频或差频,经过中频放大器和中频滤波器后,送够后端做处理。  
说的通俗点接收机就是因为天线进来的信号里面包涵每种频段的信号，但是我们只会选择其中一个信号，这里混频器的作用就是乘法。乘法得到的信号时2个信号之和和之差，相当于频谱搬移了，中频滤波器只允许一个频率过去，在经过放大和滤波就把天线的信号选择出来了。  
  
举个例子，中频滤波器的滤波频率加入是40M.为了接受天线里面的20M的信号，这里我们取差频信号，也就是说需要本振给的信号为60M的信号，这样本振信号与天线信号混频通过滤波器40M频率选择，也就是说天线接受的信号能过来的就是20M.  
60-20=40(M).这样解释很通俗，有人会说那100M的信号也会过来啊？因为100和本振60M混频也是能够得到40M通过中频滤波器的。这里其实就涉及到一个很重要的指标，镜像抑制！！  
这也是天线进来首先经过预选器的作用，预选器其实大多数就是通带滤波器，也就是主要为了限制镜像抑制的，为了不让100M对信号造成干扰，对接受的频率先进行带通选择，吧100等类似的干扰拒之门外。  
  
讲到这里你应该对接收机大致超外差有一定了解了吧。下面我们将最主要做的内容，就是设计这个本振源，只要改变这个本振源，就可以改变接受频率。  
主处理器采用TI的16位处理器MSP430F169.本振源芯片用的DDS-ad9953.中间有些高速运放采用TI的OPA690，THS3021超带宽芯片。  
  
原理图如下：  
  
  
处理器  
  
  
时钟源晶振经过倍频之后改DDS作为参考信号  
  
DDS模块  
  
  
串口通信  
  
DDS输出经过滤波器  
  
电源模块  
  
  
top层  
  
  
底层  
  
  
电源层  
  
  
地层  
  
  
  
  
3D视图。  
  
  
电源  
  
  
MSP430  
  
  
串口  
  
  
as1117  
  
  
  
  


2014-12-31 15:12 上传

[**下载附件** (169.56 KB)](http://bbs.21ic.com/forum.php?mod=attachment&aid=Mzg5OTczfDY1OTAxNGFifDE0MjIzMjM1MjF8ODIwNzg3fDg1OTE4NQ%3D%3D&nothumb=yes)

把本振源给接收机看接收频段的频谱。  
  
  
控制端主要函数如下：  
include <msp430x16x.h>  
#define CLR\_RESET   P4OUT&=~BIT0    //RESET  
#define SET\_RESET   P4OUT|=BIT0  
  
#define CLR\_IO   P4OUT&=~BIT3    //IO  
#define SET\_IO   P4OUT|=BIT3  
  
#define CLR\_SCLK   P4OUT&=~BIT1    //SCLK  
#define SET\_SCLK   P4OUT|=BIT1  
#define CLR\_SDIO   P4OUT&=~BIT2     //SDIO  
#define SET\_SDIO   P4OUT|=BIT2  
  
#define PORT\_SCLK\_OUT P4DIR |= BIT1  
#define PORT\_SDIO\_OUT P4DIR |= BIT2  
#define PORT\_SDIO\_IN  P4DIR &= ~BIT2  
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
函数名称：  WriteByte(unsigned char data)  
功   能 : （uchar data) 写数据  
参    数：输入一字节数据  
返回值  ：无  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
void WriteByte(unsigned char data)  
{  
  PORT\_SCLK\_OUT;  
  PORT\_SDIO\_OUT;  
  unsigned char i;  
  unsigned char temp;  
  temp = data;  
  for(i=0;i<8;i++)       //送入1个字节的数据  
  {  
    CLR\_SCLK;  
    nNop(4);  
    if(temp&0x80)   
      SET\_SDIO;  
    else   
      CLR\_SDIO;  
    temp<<=1;  
    nNop(2);  
    SET\_SCLK;  
    nNop(4);  
  }    
}  
  
  
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
函数名称：  AD9953\_WriteFreq(unsigned long Freq)  
功   能 :  写频率Freq转换为频率控制字写进AD9953  
           改变其输出频率，同时改变跟踪电压  
参    数：  
返回值  ：无  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
void AD9953\_WriteFreq(unsigned long Freq)  
{  
  unsigned char i;  
  unsigned long temp,FTW;  
  
  FTW = Freq\*4294967296/331775262;  //将频率转化为32位控制字，2^32=4294967296  
                                    //时钟20倍频工作，18.432MHz\*20=368640000  
  temp = FTW;  
  for(i=4;i>0;i--)  //对32位控制字进行分割  
  {  
    FTW0*=(temp&0xff);  
    temp=temp>>8;    
  }    
//  CLR\_CS;     //更新频率控制字  
  for(i=0;i<5;i++)     
  {     
     WriteByte(FTW0);     
  }     
  AD9953\_Update();     
//  SET\_CS;  
  
//  DAC0\_Update(Freq); //DAC0更新一次跟踪电压值  
}  
  
  
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
函数名称：InitAD9953()  
功   能 : 初始化  
参   数：无  
返回值  ：无  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
void InitAD9953()  
{  
    unsigned int i;    
    PORT\_SCLK\_OUT;  
    PORT\_SDIO\_OUT; //端口设置为输出模式  
    SET\_SCLK;  
//    CLR\_IOSYNC;  
    SET\_RESET;  
    nNop(100);  
    CLR\_RESET;  //初始化操作  
  
//   CLR\_CS;  //CFR1更新  
    for(i=0;i<5;i++)     
    {     
        WriteByte(CFR1);     
    }     
    AD9953\_Update();     
//    SET\_CS;    
  
//    CLR\_CS;  //CFR2更新，输出为PLL\*4=368.64MHz  
    for(i=0;i<4;i++)     
    {     
        WriteByte(CFR2);     
    }     
    AD9953\_Update();     
//    SET\_CS;  
  
//    CLR\_CS;  //ASF更新  
    for(i=0;i<3;i++)     
    {     
        WriteByte(ASF);     
    }     
    AD9953\_Update();     
//    SET\_CS;  
  
//    CLR\_CS;  //ARR更新  
    for(i=0;i<2;i++)     
    {     
       WriteByte(ARR);     
    }     
    AD9953\_Update();     
//    SET\_CS;  
  
}*