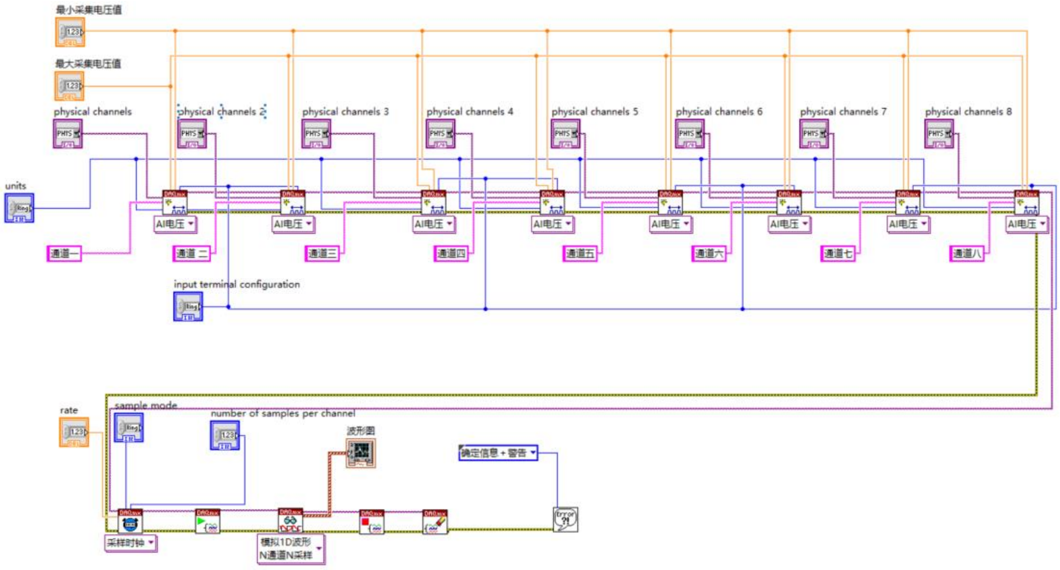
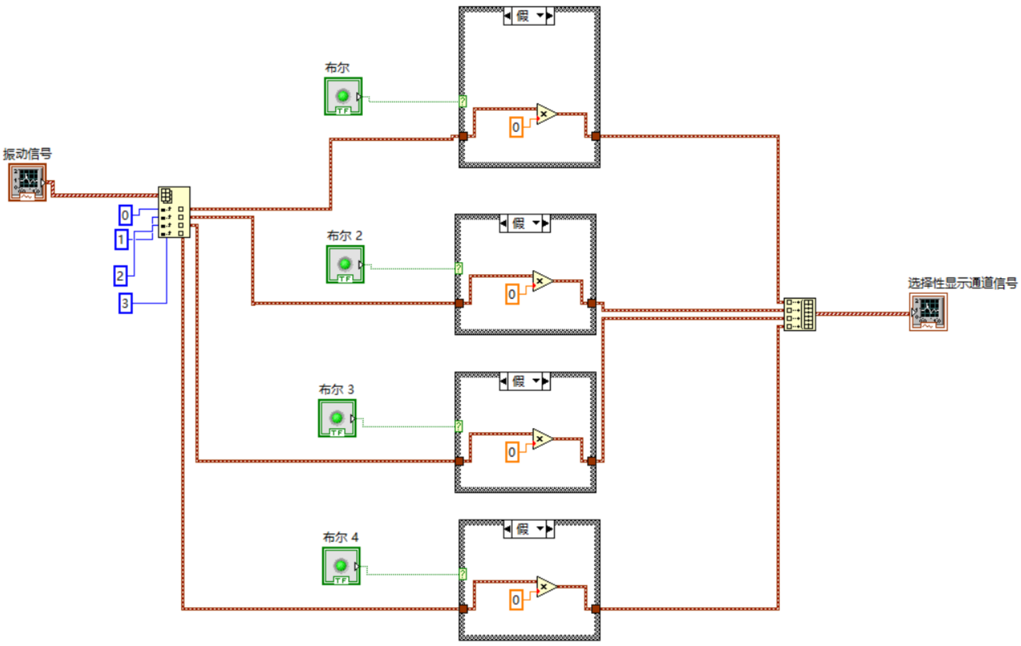
基于LabVIEW的转子动平衡测控系统软件源代码

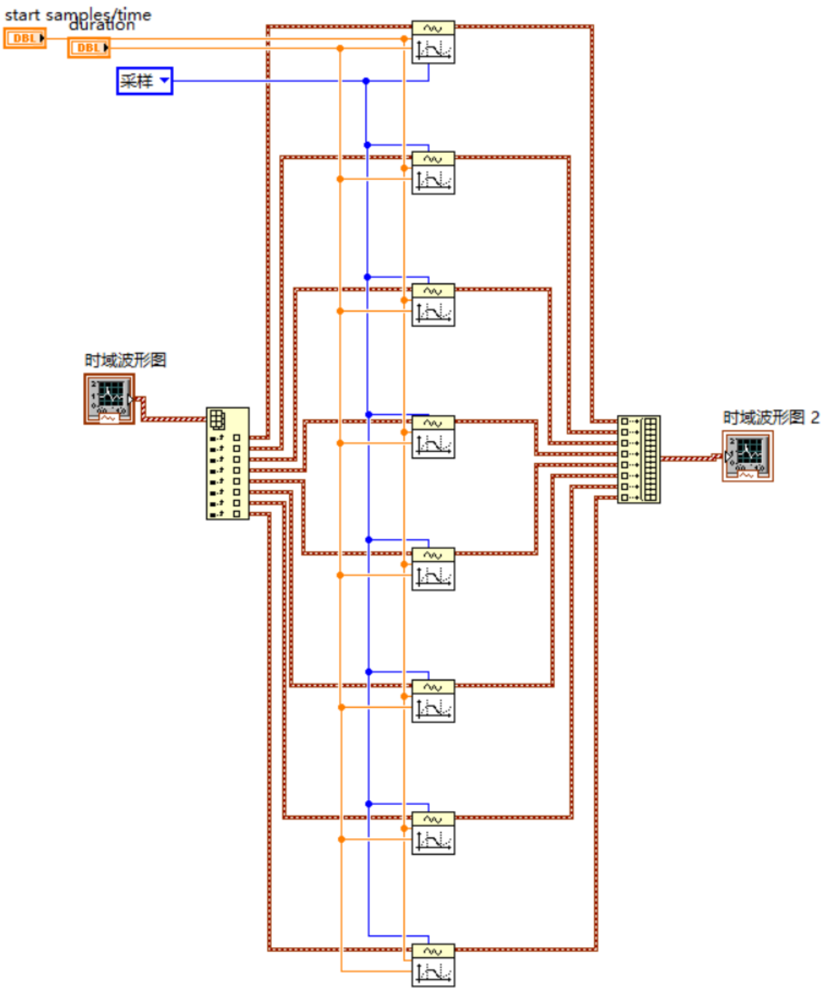
1 八通道数据采集源代码



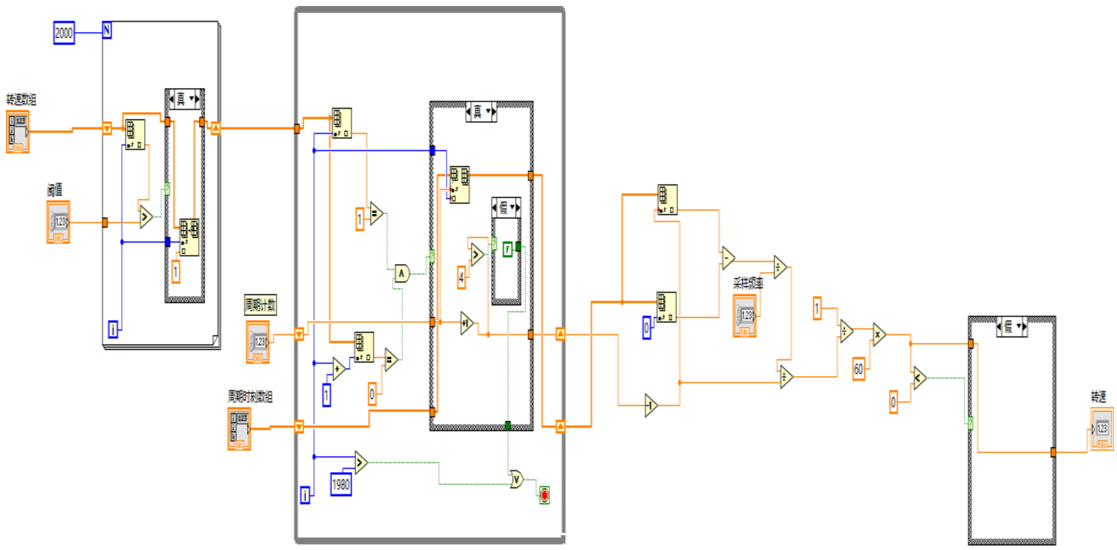
2 四通道数据选择性显示源代码



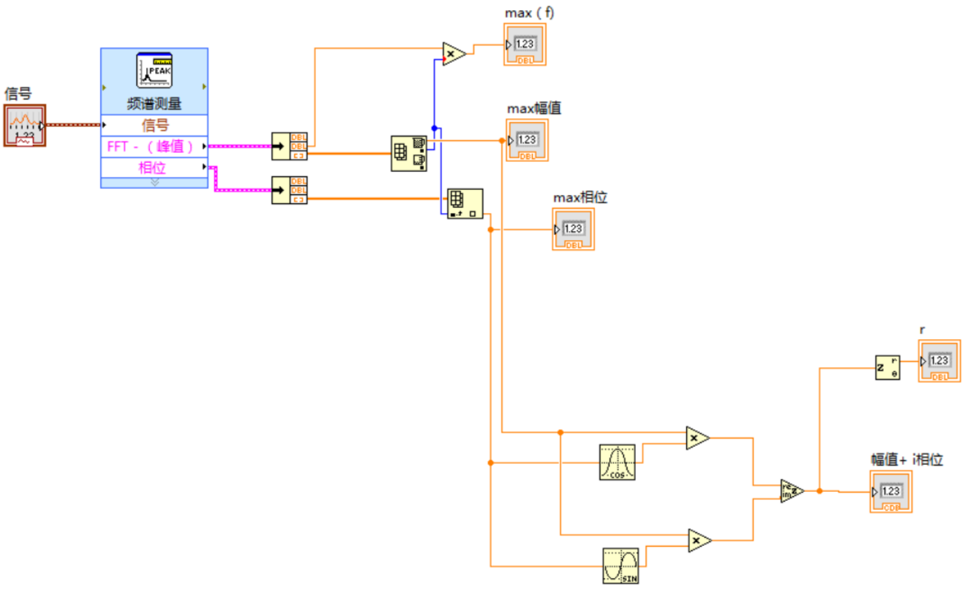
3 获取波形信号通道子集源代码



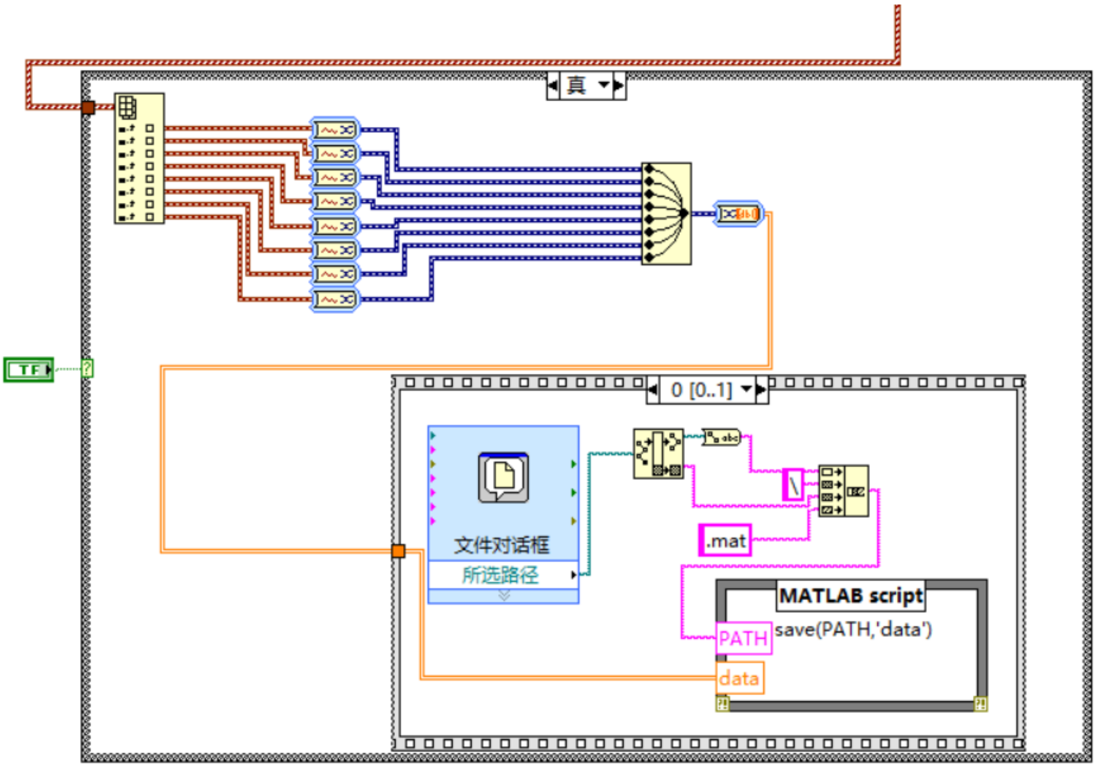
4 转速测量源代码



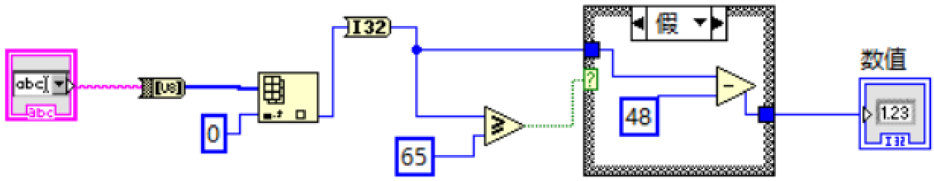
5 工频特征值提取源代码



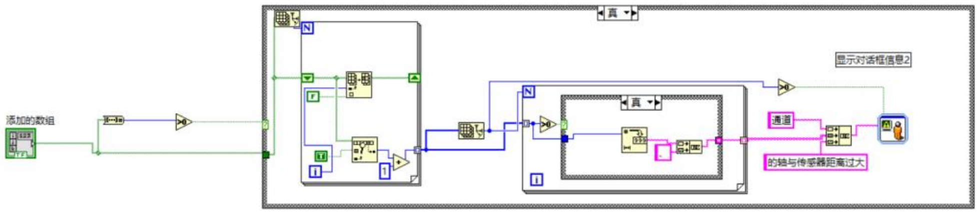
6 数据保存源代码



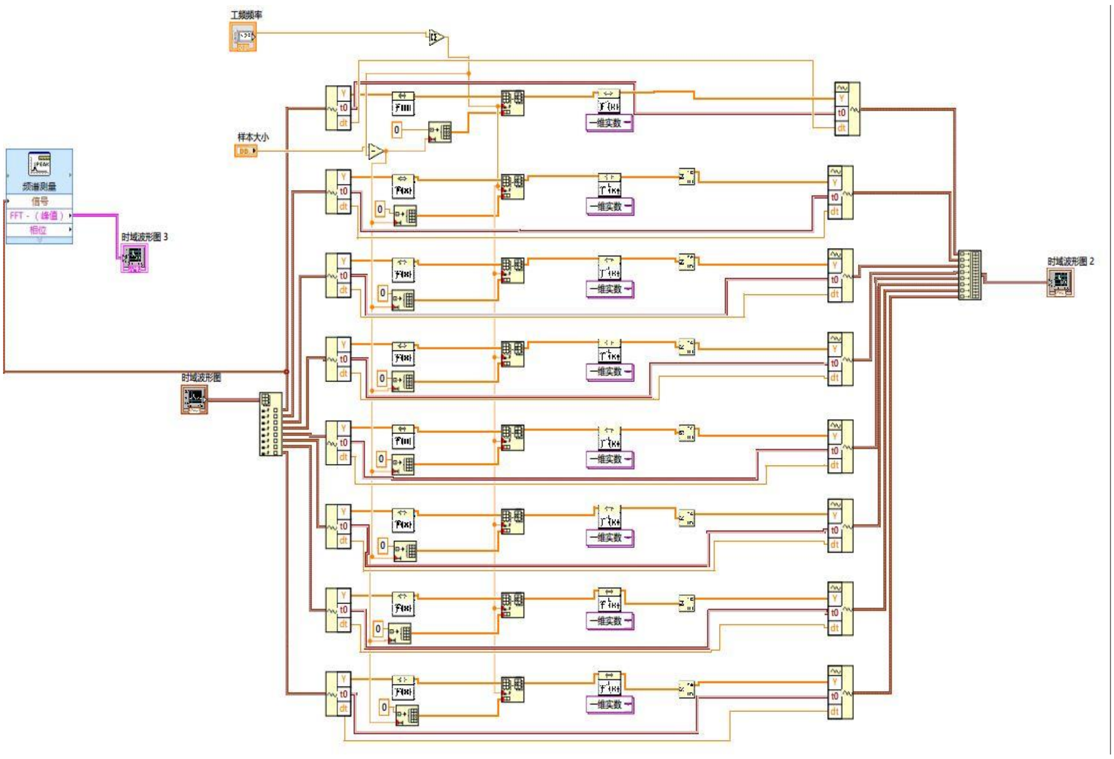
7 通道字符串转换成数值源代码



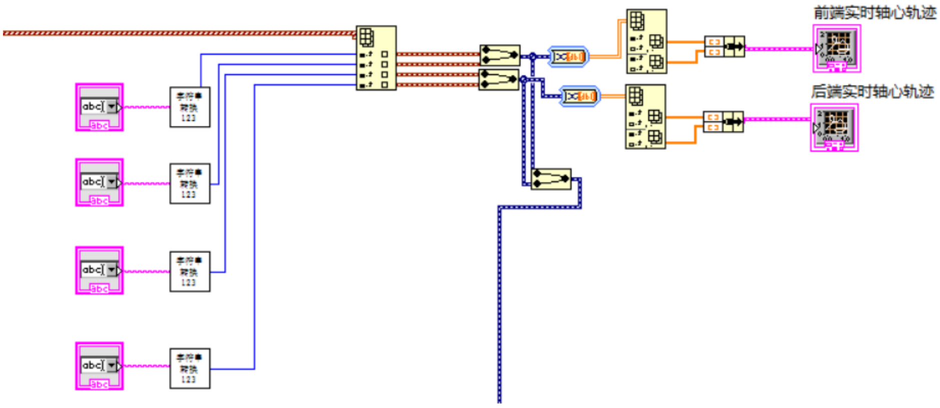
8 阈值警告源代码



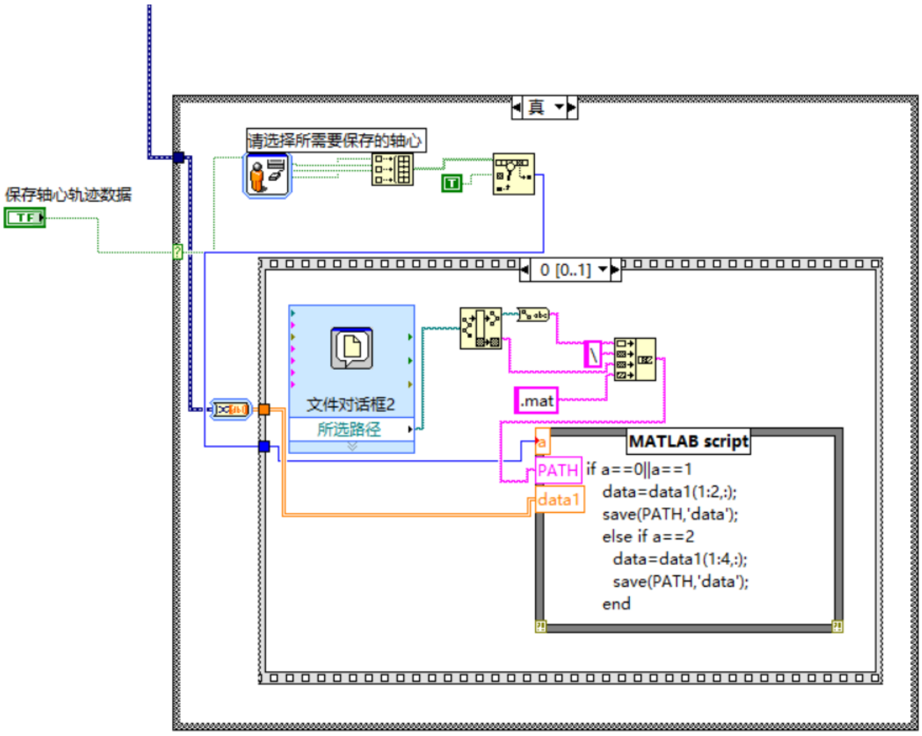
9 FFT 滤波源代码



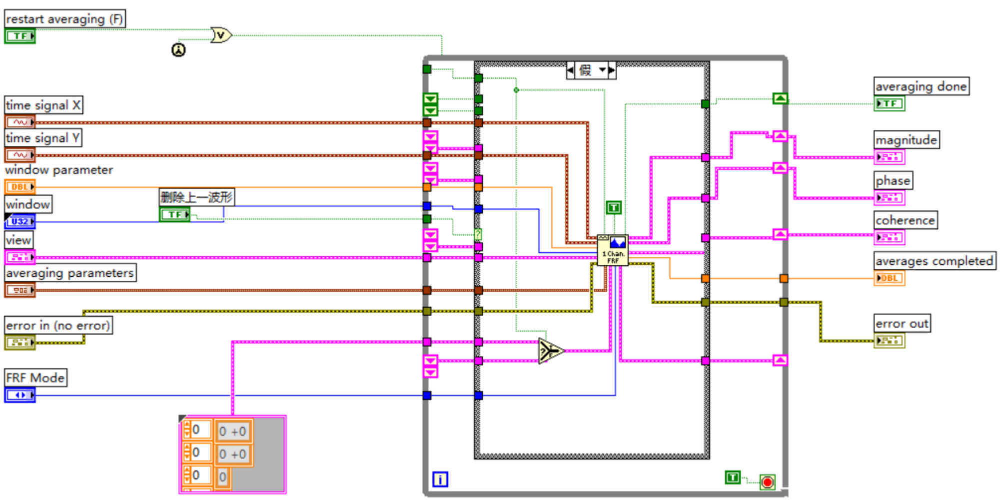
10 轴心轨迹绘制源代码



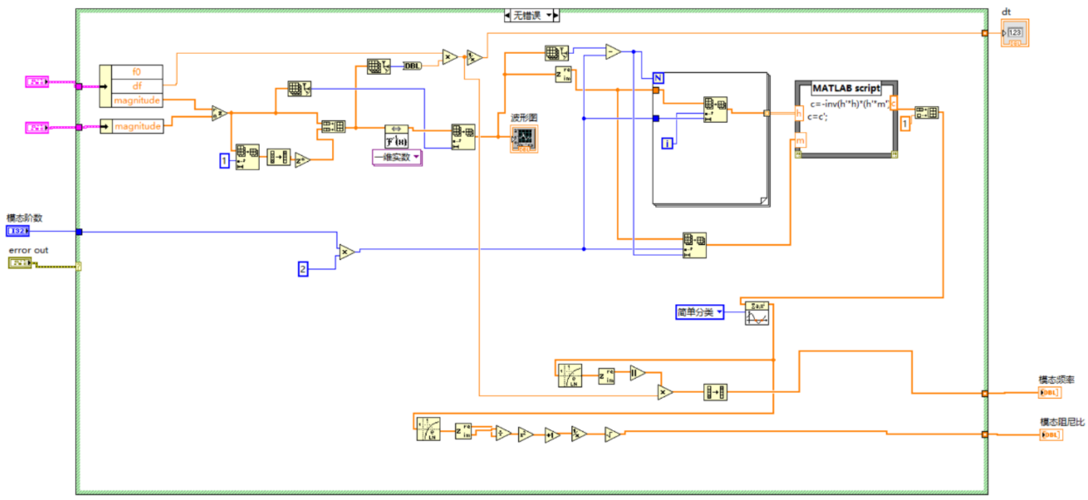
11 轴心选择性保存源代码



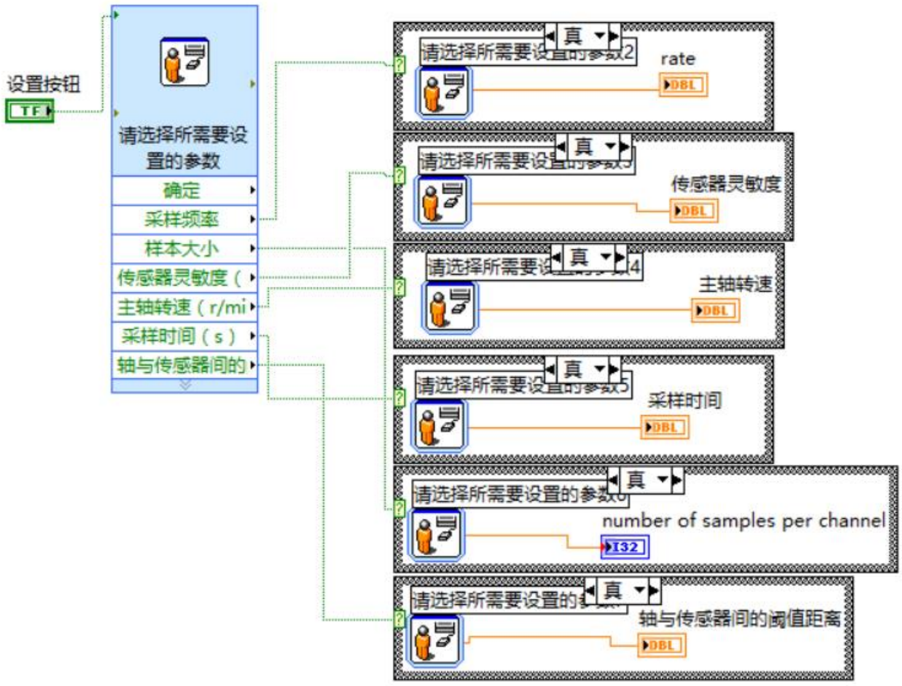
13 平均计算 FRF 源代码



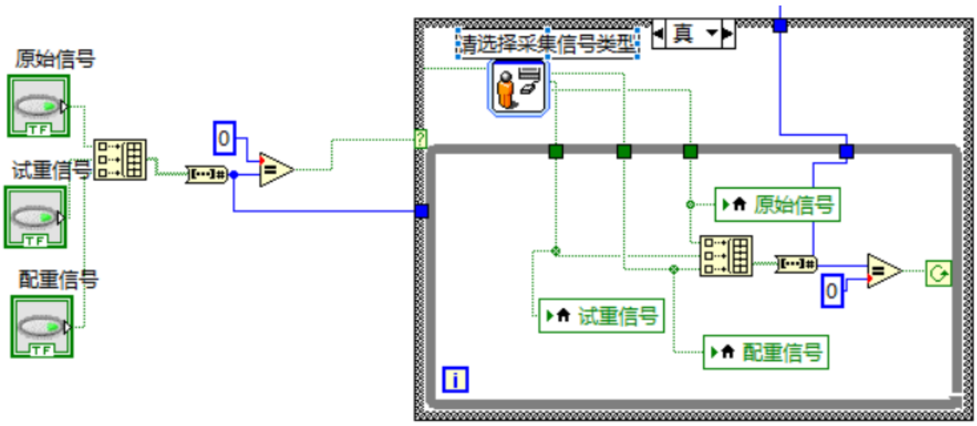
12 LSCE 法提取模态参数源代码



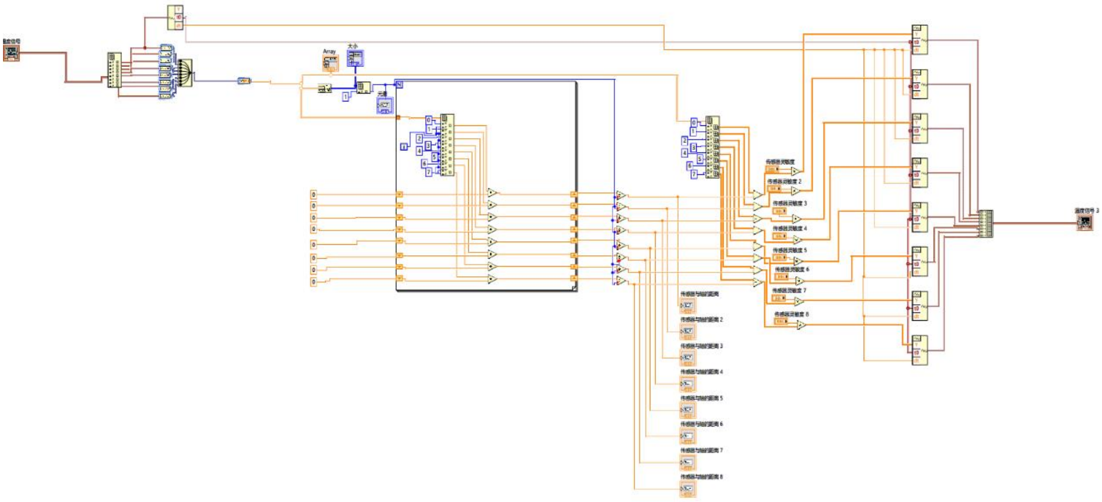
14 参数设置选择源代码



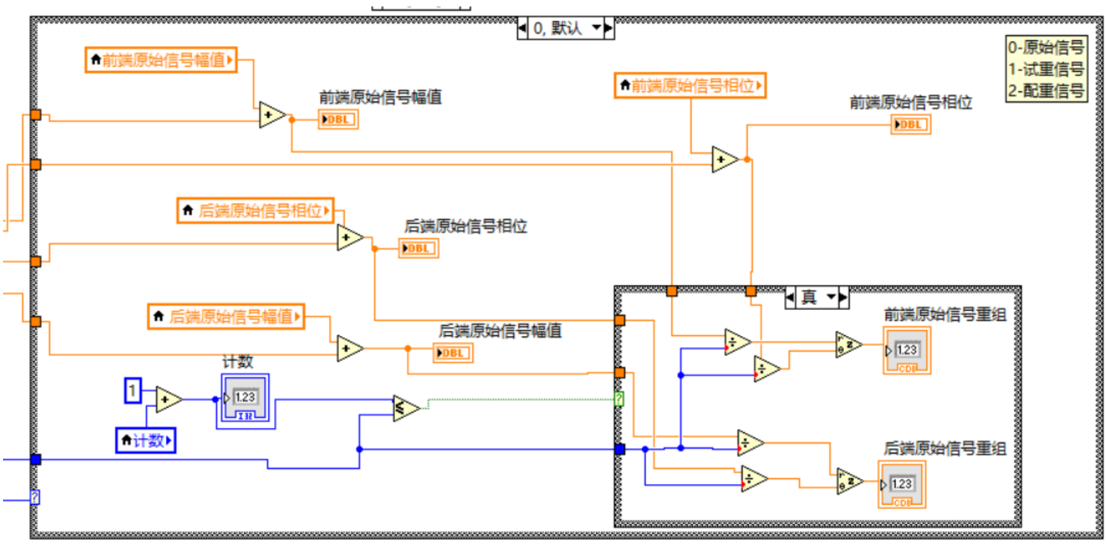
15 原始、试重、配重信号选择源代码



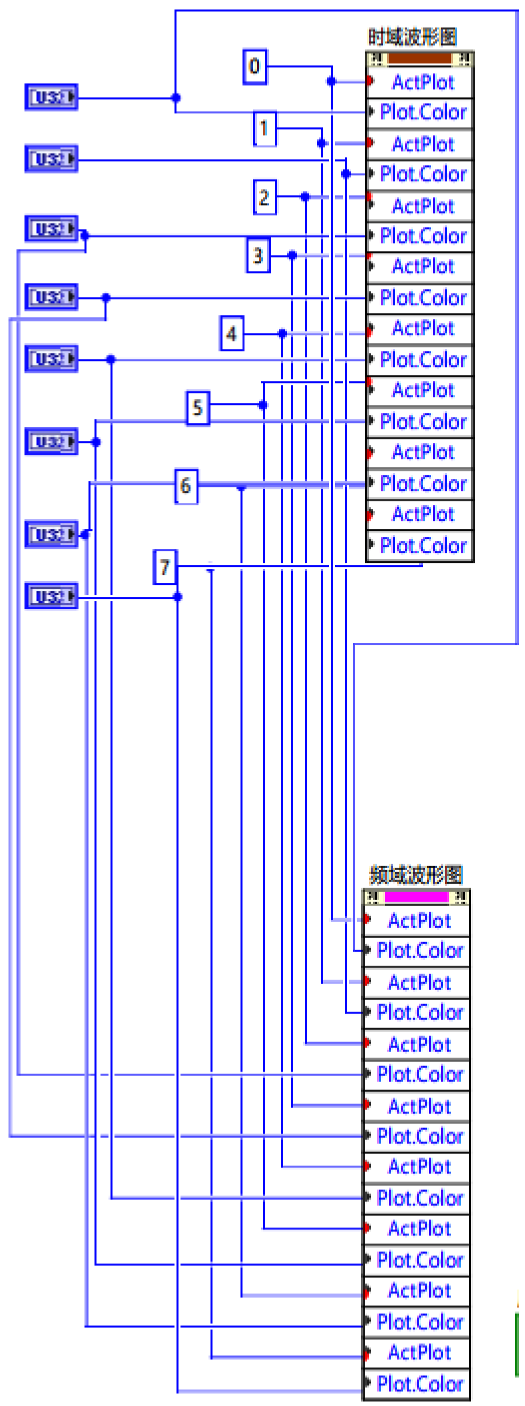
16 数据减均值及倍数放大源代码



17 原始、试重、配重振动数据多次平均并计数源代码



18 修改波形曲线颜色源代码



19 相关算法提取工频特征值源代码

```
Fs;%采样频率
Speed;%转速
frequence=Speed/60;
data=data-mean(data);
length_data=length(data);
for i=1:1:(length_data)
        x(i)=sin(2*pi*Speed/60*i/length_data);
        y(i)=cos(2*pi*Speed/60*i/length_data);
end
Rdx=0;Rdy=0;
for j=1:1:(length_data)
          m=0;n=0;
        m=data(j)*x(j);n=data(j)*y(j);
        Rdx=Rdx+m;Rdy=Rdy+n;
end
A=2*(Rdx^2+Rdy^2)^(0.5)/(length_data);
phase=atan(Rdy/Rdx);
if Rdx<0
        phase_ture=phase+pi;
else if Rdx>0&&Rdy<0
                phase_ture=phase+2*pi;
  else
              phase_ture=phase;
        end
end
```

20 影响系数法配重源代码

```
%    K = [K_ff K_fb
%             K_bf K_bb]

const=pi/180;
trialscheme1_1(1)=a1;
trialscheme1_1(2)=a2;
Holes;
%wushizhong
%4000
%vibra_f0=-56.2051-5.2217i;                          %mm
%vibra_b0=4.7116+35.0497i;

%pan1shizhong
%4000
%vibra_f1=-20.6808+2.0603i;                          %mm
%vibra_b1=-4.3554+11.7300i;

%pan4shizhong
%4000
%vibra_f2=-27.7066-18.7359i;
%vibra_b2=6.8598+39.4739i;



vibra_f0=data(1);
%vibra_b0=data(2);

vibra_f1=data(2);
%vibra_b1=data(4);

%vibra_f2=data(5);
%vibra_b2=data(6);

%Vibra0=[vibra_f0;vibra_b0];
Vibra0=vibra_f0;
%%  二次试重参数  【质量  相位  质量  相位】
 %trialscheme1_1=[0.6 8];
 %trialscheme1_2=[0.4 12];
%%     试重计算
%面 1
%  第一次试重
      me1=trialscheme1_1(1);
```

```
        angle_trial1=trialscheme1_1(2)*360/Holes;
        TW1=me1*cos(angle_trial1*const)+me1*sin(angle_trial1*const)*j;
%  第二次试重

    %    me1=trialscheme1_2(1);
    %    angle_trial1=trialscheme1_2(2)*22.5;
    %    TW2=me1*cos(angle_trial1*const)+me1*sin(angle_trial1*const)*j;
  %%  求取影响系数，K_ij，i 表示测振位置，j 表示试重次数

  K_ff=(vibra_f1-vibra_f0)/TW1;    %  第一次加试重在 f 端引起的振动
  %K_bf=(vibra_b1-vibra_b0)/TW1;    %  第一次加试重在 b 端引起的振动

  %K_fb=(vibra_f2-vibra_f0)/TW2;    %  第一次加试重在 f 端引起的振动
  %K_bb=(vibra_b2-vibra_b0)/TW2;    %  第一次加试重在 b 端引起的振动

%K=[K_ff,K_fb;K_bf,K_bb]
K=K_ff;
% CW=-inv(K)*Vibra0;
CW = -TW1*(vibra_f0/(vibra_f1-vibra_f0));
%masses=[0,0.1,0.2,0.3,0.4,0.5,0.6,0.8,1.0,1.2,1.6];
% masses=[0,0.4,0.47,0.575,0.93];
  masses=0:0.1:ceil(max(abs(CW)));
phases=(360/Holes)*const*[0:Holes-1];
%%
erros_weight_f=abs(CW(1));
w1_f=0;w2_f=0;number_h1_f=0;number_h2_f=0;
    for ii=1:5
          for jj=1:Holes
                    for kk=1:5
                          for ll=1:Holes

balance1_f=masses(ii)*cos(phases(jj))+masses(ii)*sin(phases(jj))*j;

balance2_f=masses(kk)*cos(phases(ll))+masses(kk)*sin(phases(ll))*j;
                              er_f=abs(CW(1)-balance1_f-balance2_f);
                              if er_f<erros_weight_f
                                    w1_f=masses(ii);
                                    w2_f=masses(kk);
                                    number_h1_f=jj-1;
                                    number_h2_f=ll-1;
                                    erros_weight_f=er_f;
                              elseif
er_f==erros_weight_f&w1_f+w2_f>masses(ii)+masses(kk)
                                         w1_f=masses(ii);
```

```
                        w2_f=masses(kk);
                        number_h1_f=jj-1;
                        number_h2_f=ll-1;
                        erros_weight_f=er_f;

                   end
               end
           end
      end
   end
  balance_scheme_f=[w1_f number_h1_f w2_f number_h2_f erros_weight_f];
  CW_decomp_f=balance_scheme_f


CW_decomp_f_first_weight=CW_decomp_f(1);
CW_decomp_f_first_locate=CW_decomp_f(2);
CW_decomp_f_second_weight=CW_decomp_f(3);
CW_decomp_f_second_locate=CW_decomp_f(4);



err1=CW_decomp_f(5);
```