```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define pi 3.14159265358979323846

//FYP Group: 2017-FYP-14
//Group Members: 2017-EE-106
                //2017-EE-138
                //2017-EE-142
                //2017-EE-143

int main()
{
    //Variable declaration
    double Ia,Ib,Ic,Ialpha,Ibeta,Id,Iq,Idf,theta,x,y,wm,wr,phir,slip,a;
    double Valpha,Vbeta,Vd,Vq,Vref,alpha,sum,sum1;
    double IdRef,IqRef,TeRef,phirRef,wmRef;// reference quantities
    double werror,wierror,Iderror,Idierror,Iqerror,Iqierror;
    double Rs,Rr,Lls,Llr,Lm;
    double S1,S3,S5,D1,D3,D5;
    double T0,T1,T2,Ts,Tr;
    int Vs,n,kp,ki,f,P;

    //Motor parameters
    f=50; //Supply frequency
    Rs=0.087; //Stator resistance
    Rr=0.228; //Rotor resistance
    Lls=0.0001; //Stator inductance
    Llr=0.0008; //Rotor inductance
    Lm=0.0347;  //Magnetizing inductance
    P=4; //Number of Poles
    Tr=(Llr+Lm)/Rr; //Rotor time constant

    //Input form the sensors Ia, Ib and motor mechanical speed Wm
    Ia=1;// phase a current
    Ib=2;// phase b current
    Ic=-(Ia+Ib);// phase c current
    wm=0;

    //Initial conditions
    sum=0;
    sum1=0;
    Idf=0;
    theta=0;// rotor flux angle theta
    wierror=0;
    Idierror=0;
    Iqierror=0;

    //PI gains
    kp=100;
    ki=30;

    //Reference rotor flux
    phirRef=0.96;// reference rotor flux

    //Sampling time
    Ts=0.000002;

    // Forward Clarke transform
    Ialpha=(2*Ia/3)-(Ib/3)-(Ic/3);// I alpha calculation
    Ibeta=(Ib/sqrt(3))-(Ic/sqrt(3));// I beta calculation

    //Forward park transformation
    x=cos(theta);
    y=sin(theta);
    Id=(x*Ialpha)+(y*Ibeta);//I d calculation
    Iq=(-y*Ialpha)+(x*Ibeta);// I q calculation

    //Rotor flux calculation
    //passing Id through low pass filter with time constant Tr
    Idf=sum+Ts*((Id-Idf)/Tr); //passing Id through filter
    sum=Idf;
    phir=Lm*Idf;//rotor flux

    //Rotor flux angle calculation
    wr=(Lm*Iq)/(Tr*phir);//rotor frequency calculation
    slip=(wm*P/2)+wr;// slip calculation

    //theta calculation by integrating the slip
    theta=sum1+(Ts*slip);// integration using forward Euler method
    theta=fmod(theta,2*pi);
    sum1=theta;

    //Reference Id calculation
```

```
IdRef=phirRef/Lm;

//Reference torque calculation
//implementing PI controller for speed to calculated reference torque
werror=wmRef-wm;
wierror=wierror+werror;
TeRef=(kp*werror)+(ki*wierror*Ts);

//Reference Iq calculation
IqRef=(4*TeRef*(Llr+Lm))/(3*Lm*phir);

//Implementing PI controller for Vd
Iderror=IdRef-Id;
Idierror=Idierror+Iderror;
Vd=(kp*Iderror)+(ki*Idierror*Ts);

//Implementing PI controller for Vq
Iqerror=IqRef-Iq;
Iqierror=Iqierror+Iqerror;
Vq=(kp*Iqerror)+(ki*Iqierror*Ts);

//Inverse park transform
Valpha=(x*Vd)-(y*Vq);// V alpha calculation
Vbeta=(y*Vd)+(x*Vq);// V beta calculation

//space vector modulation

//Vref and angle calculation
Vref=sqrt(pow(Valpha,2)+pow(Vbeta,2));
alpha=atan2(Vbeta,Valpha);
if(alpha<=0)
{
    alpha=alpha+(2*pi);
}

//Sector number calculation
n=floor(alpha/(pi/3))+1;

//Modulation index calculation
a=(Vref)/((2/3)*Vs);
//modulation index can not be greater than 0.866 to avoid negative time
if(a>=0.86)
{
    a=0.86;
}

//Switching Times calculation
T1 = Ts*a*sin((n*pi/3)-alpha)/sin(pi/3);
T2 = Ts*a*sin(alpha-((n-1)*pi/3))/sin(pi/3);
T0 = Ts-T1-T2;

//Time calculation for high side switches
switch (n)
{
case 1:
S1 = T1 + T2 + T0/2;
S3 = T2 + T0/2;
S5 = T0/2;
break;
case 2:
S1 = T1 + T0/2;
S3 = T1 + T2 + T0/2;
S5 = T0/2;
break;
case 3:
S1 = T0/2;
S3 = T1+ T2 + T0/2;
S5 = T2 + T0/2;
break;
case 4:
S1 = T0/2;
S3 = T1 + T0/2;
S5 = T1 + T2 + T0/2;
break;
case 5:
S1 = T2 + T0/2;
S3 = T0/2;
S5 = T1 + T2 + T0/2;
break;
case 6:
S1 = T1 + T2 + T0/2;
S3 = T0/2;
S5 = T1 + T0/2;
break;
```

```
    }

    //Duty Cycles calculation
    D1 = S1/Ts;
    D3 = S3/Ts;
    D5 = S5/Ts;

    return 0;
}

```