

## 10. Anexo A

```
***** Archivos de cabecera *****

#include "math.h"
#include "stdio.h"

***** Definiciones propias *****

#define TM (*DELT) //Tiempo de muestreo en segundos
#define FREC 10000
#define Ts (1./FREC) //Periodo de sampleo en seg
#define Tmed 1.14
#define CONTADOR_s 1./(FREC*TM)
#define M_PI 3.1415926535897932384626433832795
#define TAMSENO 200 //Generamos un seno de 200 puntos
#define contador_med Tmed/Ts
#define inv_contador Ts/Tmed

***** Para nombrar como PSCAD (Fortran convention) *****

typedef double real;
typedef long int integer;
typedef long int logical;

void cpll_(real *vap,real *vbp,real *vcp,real *bg,real *brp,real *varp,real *vbrp,real
*vcrp,real *Deb,real *DELT,logical *TIMEZERO,real *TIME)

{
    //Variables para realizar el algoritmo

    static int contador = 0; //Contador de ciclos

    //Variables para obtener la salida

    static float U2an = 0.; //Tensión de salida del oscilador
    static float U2bn = 0.;
    static float U2cn = 0.;

    //Variables del Detector de fases

    static float Udan = 0; // Tensión de salida del Multiplicador de Frecuencias
    static float Udbn = 0;
    static float Udcn = 0;
    static float U1an = 0; //Muestreo de la señal de la fase A (referencia)
    static float U1bn = 0;
```



```
static float U1cn = 0;
static float U1anant = 0; // Variable intermedia. Guarda el valor anterior del
muestreo de tensión
static float U1bnant = 0;
static float U1cnant = 0;
static float Udanant; // Tensión de entrada proveniente del Multiplicador en el
instante anterior
static float Udbnant;
static float Udcnant;
static float Kd; // Ganancia del multiplicador

// Variables del Filtro

static float Ufan = 0.; // Tensión de salida del filtro
static float Ufbn = 0.;
static float Ufcn = 0.;
static float Ufanant= 0.; // Tensión de salida del filtro en el instante anterior
static float Ufbnант= 0.;
static float Ufcnант= 0.;
static float bo = 0.; // Coeficiente de Ud(n)
static float b1 = 0.; // Coeficiente de Ud(n-1)
static float a1 = 0.; // Coeficiente de Uf(n-1)
static float Tau1; // Proporcional del PI
static float Tau2; // Integral del PI

// Variables para el Oscilador

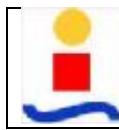
static float Fi2a =0.; // Valor de la fase a la salida del oscilador
static float Fi2b =0.;
static float Fi2c =0.;
static float w0= 50.; // Frecuencia central del oscilador
static float w2a= 0.; // Frecuencia de la fase A calculada por el PLL
static float w2b= 0.;
static float w2c= 0.;
static float w2auxa= 0.; // Frecuencia de la fase A empleada en caso de fallos
static float w2auxb= 0.;
static float w2auxc= 0.;
static float w2ma= 0.; // Frecuencia media de la fase A
static float w2mb= 0.;
static float w2mc= 0.;
static float w2maaux= 0.; // Frecuencia media de la fase A empleada en caso de
fallos
static float w2mbaux= 0.;
static float w2mcaux= 0.;
static float fa= 0.; // Flag de fallo en la fase A
static float fb= 0.;
static float fc= 0.;
static float ff= 0.; // Flag de inicio de cálculo de frecuencia media
```



## Proyecto Fin de Carrera

### Diseño de un PLL para el inversor de un sistema de alimentación ininterrumpida (SAI)

```
static int fp=0; //Flag de indicador de estado  
static int rs=0; // Flag de inicio de sincronización con la red  
static int rsa=0; // Flag indicador de sincronización alcanzada  
static int rsb=0;  
static int rsc=0;  
static float Ko; // Ganancia del Oscilador  
  
//Variables para la generación del seno a la salida  
  
static int indiceg = 0.; //Índice para generar los 3 senos  
static float indicera = 0.; //Índice para obtención del valor del seno en la fase A  
static float indicerb = 0.;  
static float indicerc = 0.;  
static float incrementoa = 0.; //variable para marcar cada cuantos puntos  
tomamos un valor del seno  
static float incrementob = 0.;  
static float incrementoc = 0.;  
static int indinta = 0.; //Índice con el que entramos en la tabla correspondiente a  
la fase A  
static int indintb = 0.;  
static int indintc= 0.;  
static float indiceraaux = 0.;  
static float indicerbaux = 0.;  
static float indicercaux = 0.;  
static float incrementoaaux = 0.;  
static float incrementobaux = 0.;  
static float incrementocaux = 0.;  
static int indinttaaux = 0.;  
static int indintbaux = 0.;  
static int indintcaux= 0.;  
static float salidaa = 0.; //Onda generado por el PLL para la fase A  
static float salidab = 0.;  
static float salidac = 0.;  
static float salidaaaux = 0.; //Onda auxiliar generado por el PLL para la fase A  
static float salidabaux = 0.;  
static float salidaaux = 0.;  
static float salidanta = 0.; //Onda generado por el PLL para la fase A en el  
instante anterior  
static float salidantb = 0.;  
static float salidantc = 0.;  
static float aa = 0.; //Diferencia entre la red y la onda de la fase A en el paso  
por cero positivo  
static float ab = 0.;  
static float ac = 0.;  
static float senoa[TAMSENO]; //Tabla con el seno de la onda de la fase A  
static float senob[TAMSENO];  
static float senoc[TAMSENO];
```



## Proyecto Fin de Carrera

### Diseño de un PLL para el inversor de un sistema de alimentación ininterrumpida (SAI)

```
static int contMed=0; /*Contador para el cálculo de la frecuencia media
static int f=0.; //Indicador de que el primer cálculo de la media de la frecuencia
ha sido realizado
/* Usamos esta variable lógica para que estos */
/* cálculos sólo se realicen una vez y acelere */
/* el proceso de simulación. */

if(*TIMEZERO)
{
    contador= 0;
    //Constantes del PI del Filtro

    Tau1 = 0.001798268;//798268;
    Tau2 = 0.019098597;
    a1= -1.;

    bo= (Ts/2*Tau1)*(1+(1/tan(Ts/(2*Tau2))));
    b1= (Ts/2*Tau1)*(1-(1/tan(Ts/(2*Tau2))));
    Ko= 2.424242;
    Kd= 146.422;

    //Inicialización de las variables del algoritmo

    U2an= 0.; U2bn= 0.; U2cn= 0.;
    Udanant= 0.; Udbnnt= 0.; Udcnnt= 0.;
    Ufan= 0.; Ufbn= 0.; Ufcn= 0.;
    Ufannt= 0.; Ufbnnt= 0.; Ufcnnt= 0.;
    Fi2a= 0.; Fi2b= 0.; Fi2c= 0.;

    w0=50*2*M_PI;
    f=0;
    ff=0;
    fa=0;
    fb=0;
    fc=0;
    fp=0;
    i=0;
    rs=0;
    rsa=0; rsb=0; rsc=0;
    indicerc=133.3;
    indintc=133;
    indicerc=66.6;
    indintc=66;

    // Calculamos la tabla del seno

    for(indiceg=0;indiceg<TAMSENO;indiceg++)
    {
        senoa[indiceg] = sqrt(2.)*230*sin(2.*M_PI*indiceg/TAMSENO-0*M_PI/3);
    }
}
```



## Proyecto Fin de Carrera

### Diseño de un PLL para el inversor de un sistema de alimentación ininterrumpida (SAI)

```
senob[indiceg] = sqrt(2.)*230*sin(2.*M_PI*indiceg/TAMSENO-2*M_PI/3);  
senoc[indiceg] = sqrt(2.)*230*sin(2.*M_PI*indiceg/TAMSENO-4*M_PI/3);  
}
```

```
indiceg = 0;
```

```
}
```

```
contador++;
```

```
if (contador>=CONTADOR_s)
```

```
{
```

```
contador=0;
```

*//Traspaso a variables locales*

```
U1an = 1000.*(*vap);  
U1bn = 1000.*(*vbp);  
U1cn = 1000.*(*vcp);
```

*//Cálculo de la señal del multiplicador*

```
Udan= Kd*U1an*U2an;  
Udbn= Kd*U1bn*U2bn;  
Udcn= Kd*U1cn*U2cn;
```

*//Cálculo de la señal del filtro*

```
Ufan = -a1*Ufanant+bo*Udan+b1*Udanant;  
Ufbn = -a1*Ufbn+bo*Udbn+b1*Udbnant;  
Ufcn = -a1*Ufcn+bo*Udcn+b1*Udcnant;
```

*//Cálculo de la fase Fi2 y la frecuencia w2*

```
Fi2a= Fi2a+(w0+Ko*Ufan)*Ts;  
w2a=(w0+Ko*Ufan)/(2*M_PI);
```

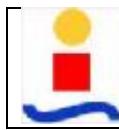
```
Fi2b= Fi2b+(w0+Ko*Ufbn)*Ts;  
w2b=(w0+Ko*Ufbn)/(2*M_PI);
```

```
Fi2c= Fi2c+(w0+Ko*Ufcn)*Ts;  
w2c=(w0+Ko*Ufcn)/(2*M_PI);
```

*//Calculamos la señal de salida*

```
if (Fi2a>M_PI) Fi2a=Fi2a-2*M_PI;
```

```
if(Fi2a>=0)
```



## Proyecto Fin de Carrera

### Diseño de un PLL para el inversor de un sistema de alimentación ininterrumpida (SAI)

```
{  
    U2an=1;  
}  
else  
{  
    U2an=-1;  
}  
  
if (Fi2b>M_PI) Fi2b=Fi2b-2*M_PI;  
  
if(Fi2b>=0)  
{  
    U2bn=1;  
}  
else  
{  
    U2bn=-1;  
}  
  
if (Fi2c>M_PI) Fi2c=Fi2c-2*M_PI;  
  
if(Fi2c>=0)  
{  
    U2cn=1;  
}  
else  
{  
    U2cn=-1;  
}  
  
//Calculamos la frecuencia media obtenida  
  
if(contMed>=contador_med)  
{  
    contMed=0;  
    w2ma=w2auxa*inv_contador;  
    w2auxa=0;  
    w2mb=w2auxb*inv_contador;  
    w2auxb=0;  
    w2mc=w2auxc*inv_contador;  
    w2auxc=0;  
    if(f==0) f=1;  
}  
else  
{  
    contMed++;  
    w2auxa+= w2a;  
    w2auxb+= w2b;
```



## Proyecto Fin de Carrera

### Diseño de un PLL para el inversor de un sistema de alimentación ininterrumpida (SAI)

```
w2auxc+= w2c;
```

```
}
```

```
//Disparamos flags de alerta ante fallos en la red
```

```
if(f==1)
```

```
{
```

```
fp=1;
```

```
if(w2ma>=50.55) fa=1;
```

```
if(w2ma<=49.45) fa=2;
```

```
if(w2mb>=50.55) fb=1;
```

```
if(w2mb<=49.45) fb=2;
```

```
if(w2mc>=50.55) fc=1;
```

```
if(w2mc<=49.45) fc=2;
```

```
if(((w2ma>49.45)&&(w2ma<50.55))&&((w2mb>49.45)&&(w2mb<50.55))&&  
((w2mc>49.45)&&(w2mc<50.55)))
```

```
{
```

```
ff=0;fa=0;fb=0;fc=0;
```

```
}
```

```
}
```

```
if(((fa>=1)|| (fb>=1)|| (fc>= 1)|| (*bg)==1))&&(f==1))
```

```
{
```

```
if(ff==0)
```

```
{
```

```
ff=1;
```

```
if((fa==1)|| (fb==1)|| (fc==1))
```

```
{
```

```
w2maaux=50.5;
```

```
w2mbaux=50.5;
```

```
w2mcaux=50.5;
```

```
rs=1;
```

```
}
```

```
if((fa==2)|| (fb==2)|| (fc==2))
```

```
{
```

```
w2maaux=49.5;
```

```
w2mbaux=49.5;
```

```
w2mcaux=49.5;
```

```
rs=1;
```

```
}
```



## Proyecto Fin de Carrera

### Diseño de un PLL para el inversor de un sistema de alimentación ininterrumpida (SAI)

}

```
if(w2maaux>=50) w2maaux=w2maaux-.00005;  
if(w2maaux<50) w2maaux=w2maaux+.00005;  
if(w2mbaux>=50) w2mbaux=w2mbaux-.00005;  
if(w2mbaux<50) w2mbaux=w2mbaux+.00005;  
if(w2mcaux>=50) w2mcaux=w2mcaux-.00005;  
if(w2mcaux<50) w2mcaux=w2mcaux+.00005;
```

//Calculamos seno de salida con puntero marcado por la frecuencia del PLL (fallo)//

```
incrementoa = (200*w2maaux)/10000;  
indicera+= incrementoa;  
indinta = (int)(indicera);
```

```
if (indicera>=200)  
{  
    indinta-=200;  
    indicera-=200.;  
}
```

```
salidaa=senoa[indinta];
```

```
incrementob = (200*w2mbaux)/10000;  
indicerb+= incrementob;  
indintb = (int)(indicerb);
```

```
if (indicerb>=200)  
{  
    indintb-=200;  
    indicerb-=200.;  
}
```

```
salidab=senob[indintb];
```

```
incrementoc = (200*w2mcaux)/10000;  
indicerc+= incrementoc;  
indintc = (int)(indicerc);
```

```
if (indicerc>=200)  
{  
    indintc-=200;  
    indicerc-=200.;  
}
```

```
salidac=senoc[indintc];
```

}



## Proyecto Fin de Carrera

Diseño de un PLL para el inversor de un sistema de alimentación ininterrumpida (SAI)

else

{

*//Calculamos el seno de salida con referencia la red (red OK)*

```
fa=0;  
fb=0;  
fc=0;  
ff=0;  
fp=0;
```

*// Preparamos el PLL para la sincronización con la red*

```
if(rs==1)  
{
```

fp=2;

*// Sincronización en frecuencia*

```
if(w2ma>w2maaux) w2maaux=w2maaux+0.0001;  
if(w2ma<=w2maaux) w2maaux=w2maaux-0.0001;  
if(w2mb>w2mbaux) w2mbaux=w2mbaux+0.0001;  
if(w2mb<=w2mbaux) w2mbaux=w2mbaux-0.0001;  
if(w2mc>w2mcaux) w2mcaux=w2mcaux+0.0001;  
if(w2mc<=w2mcaux) w2mcaux=w2mcaux-0.0001;
```

*// Sincronización en fase*

```
incrementoa = (200*w2maaux)/10000;
```

```
indicera+= incrementoa;
```

```
indinta = (int)(indicera);
```

```
if ((U1anant<0)&&(U1an>0))
```

{

aa=salidaa-U1an;

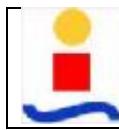
```
if((aa>5)|| (aa<-5))  
{
```

```
if(aa<=0)
```

{

indicera++;

indinta++;



## Proyecto Fin de Carrera

### Diseño de un PLL para el inversor de un sistema de alimentación ininterrumpida (SAI)

```
        }
```

```
        if(aa>0)
        {
            indicera--;
            indinta--;
        }
        else rsa=1;
    }

if (indicera>=200)
{
    indinta-=200;
    indicera-=200.;
}

salidaa=senoa[indinta];

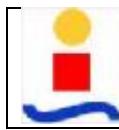
incrementob = (200*w2mbaux)/10000;
indicerb+= incrementob;
indintb = (int)(indicerb);

if ((U1bnant<0)&&(U1bn>0))
{
    ab=salidab-U1bn;

    if((ab>5)|| (ab<-5))
    {

        if(ab<=0)
        {
            indicerb++;
            indintb++;
        }
        if(ab>0)
        {
            indicerb--;
            indintb--;
        }
    }
    else rsb=1;
}

if (indicerb>=200)
{
```



## Proyecto Fin de Carrera

### Diseño de un PLL para el inversor de un sistema de alimentación ininterrumpida (SAI)

```
indintb=200;
indicerb=200.;
}

salidab=senob[indintb];

incrementoc = (200*w2mcaux)/10000;
indicerc+= incrementoc;
indintc = (int)(indicerc);

if ((U1cnant<0)&&(U1cn>0))
{
    ac=salidac-U1cn;

    if((ac>5)|| (ac<-5))
    {

        if(ac<=0)
        {
            indicerc++;
            indintc++;
        }
        if(ac>0)
        {
            indicerc--;
            indintc--;
        }
    }
    else rsc=1;
}

if (indicerc>=200)
{
    indintc-=200;
    indicerc-=200.;
}

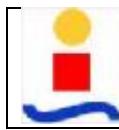
salidac=senoc[indintc];

}

if((rsa==1)&&(rsb==1)&&(rsc==1)) rs=0;

// Cuando el PLL está sincronizado en fase y frecuencia con la Red reenganchamos//

if (rs==0)
{
```



## Proyecto Fin de Carrera

### Diseño de un PLL para el inversor de un sistema de alimentación ininterrumpida (SAI)

```
rsa=0;
rsb=0;
rsc=0;
incrementoa = (200*w2ma)/10000;
indicera+= incrementoa;
indinta = (int)(indicera);
if(f==0)
{
}
else
{
    if ((U1an<0)&&(U1an>0))
    {
        aa=salidaa-U1an;
        if((aa>5)|| (aa<-5))
        {
            if(aa<=0)
            {
                indicera++;
                indinta++;
            }
            if(aa>0)
            {
                indicera--;
                indinta--;
            }
        }
    }
    if(indicera>=200)
    {
        indinta-=200;
        indicera-=200.;
    }
    salidaa=senoa[indinta];
}

incrementob = (200*w2mb)/10000;
indicerb+= incrementob;
indintb = (int)(indicerb);
```



## Proyecto Fin de Carrera

Diseño de un PLL para el inversor de un sistema de alimentación ininterrumpida (SAI)

```
if ((U1bnant<0)&&(U1bn>0))
{
    ab=salidab-U1bn;
    if((ab>5)|| (ab<-5))
    {
        if(ab<=0)
        {
            indicerb++;
            indintb++;
        }
        if(ab>0)
        {
            indicerb--;
            indintb--;
        }
    }
}

if (indicerb>=200)
{
    indintb-=200;
    indicerb-=200.;
}

salidab=senob[indintb];
incrementoc = (200*w2mc)/10000;
indicerc+= incrementoc;
indintc = (int)(indicerc);

if ((U1cnant<0)&&(U1cn>0))
{
    ac=salidac-U1cn;
    if((ac>5)|| (ac<-5))
    {
        if(ac<=0)
        {
            indicerc++;
            indintc++;
        }
        if(ac>0)
        {
            indicerc--;
            indintc--;
        }
    }
}
```



## Proyecto Fin de Carrera

### Diseño de un PLL para el inversor de un sistema de alimentación ininterrumpida (SAI)

}

}

```
if (indicerc>=200)
{
    indintc-=200;
    indicerc-=200.;
}
```

```
    salidac=senoc[indintc];
```

```
}
}
```

}

*//Actualizamos las variables*

```
Udanant=Udan;
Ufanant=Ufan;
salidanta=salidaa;
U1anant=U1an;
```

```
Udbnnt=Udbn;
Ufbnnt=Ufbn;
salidantb=salidab;
U1bnant=U1bn;
```

```
Udcnnt=Udcn;
Ufcnnt=Ufcn;
salidantc=salidac;
U1cnnt=U1cn;
}
```

*//Se asignan los valores de salida*

```
*varp=salidaa;
*vbrp=salidab;
*vcrp=salidac;
```

}