

B. CODIGO DE LA IMPLEMENTACION

Este anexo muestra la implementación del bloque AMC, el cual se desarrolló enteramente en el lenguaje C, el cual es el corazón de Matlab® por lo que permite utilizar sus librerías y funciones, lo que es muy útil cuando se quiere realizar un proceso ágil entre código fuente y bloques de Simulink® ya construidos.

A continuación se hace una breve explicación del código que maneja la estructura y las funcionalidades del bloque AMC.

Aquí se definen los valores de G y BW, G=1/8 definido por el estándar para WiMAX móvil y BW=2.5MHz, escogemos este BW para que su procesamiento sea más rápido, y según el estándar para BW= 2.5MHz la FFT correspondiente es de 256

```
disp('Ancho de Banda a utilizar es de 2.5 MHz para una FFT de 256');
disp('El prefijo cíclico es de 1/8');%definido por el estándar
G= 1/8;
BW=2.5; %Se presenta información pertinente sobre el BW y prefijo
cíclico a utilizar.
```

Aquí se define un vector con las frecuencias de los canales y con factores de sobre muestreo cada valor es asociado al canal que se vaya a usar.

```
channels=[1.75 1.5 1.25 2.75 2.0];%
oversampling=[8/7 86/75 144/125 316/275 57/50 8/7];%
```

Se define el tamaño de la transformada de Fourier, número de portadoras usadas para datos, frecuencia de espaciamiento, el tiempo de guarda, el tiempo de símbolo y el de muestreo.

```
Nused=200; Nfft=256;
fs=(floor((n*BW*1e6)/8000))*8000; %frecuencia de muestreo
freqspacing= fs/Nfft; %espaciamiento de frecuencia
Tb= 1/freqspacing; %tiempo útil de símbolo
Tg= G*Tb ;%tiempo de guarda
Ts=Tb+Tg ;%tiempo de símbolo
samplingtime= Tb/Nfft; %tiempo de muestreo
```

Aquí se genera un vector de 5 elementos, calcula el residuo de la división entre el BW y cada frecuencia del vector channels.

```
for i=1:5%
y(i)=rem(BW,channels(i));%

if y(i)==0 % inicializa N con el ultimo valor de sobre-muestreo,
resultado del cálculo anterior
    n=oversampling(i);
end
end

y=(y(1))*(y(2))*(y(3))*(y(4))*(y(5)); % Se utiliza para determinar si
algún residuo de la división del BW y de los canales dio 0.
if y~=0
```

Si ninguno de los residuos es 0 entonces usa sobre-muestreo de 8/7

```
n=8/7;  
end
```

Ahora se crea el polinomio generador, a partir de un campo de galois el cual se utiliza en el codificador Reed-Solomon

```
genpoly=gf(1,8);  
for idx=0:15  
    genpoly=conv(genpoly,[1 gf(2,8)^idx]);  
end
```

Este es el vector primitivo para IEEE802.16
primepoly=[1 0 0 0 1 1 1 0 1];

Aquí se genera la estructura de Trellis, la cual permite realizar una óptima codificación convolucional que se utiliza en el codificador Reed Solomon.

```
convvec=poly2trellis(7,[171,133]);
```

La siguiente parte del código inicializa los parámetros para la codificación, ahí es donde se selecciona dependiendo del valor de cSNR, como se va a modular la señal solo que cambia los valores y características de funcionalidad en; el tamaño de la trama entrada, en el generador de datos, "reshape", cambios en el codificador RS y CC, el tipo de modulación y codificación.

Determina que tipo de modulación y codificación debe usarse para un determinado valor de SNR, es el objetivo del bloque AMC

```
if (6.4<=cSNR&cSNR<9.4)  
    inputsize=88;  
    seqafterrand=inputsize+8;  
    shortening=[1:12];  
    shorteningRx=[1:11];  
    punvec=reshape([1, 1],2,1);%rata de convolución 1/2  
    Ncbps=192;%selector de RS 12*8(24*8)  
    k=0:Ncbps-1;  
    mk=(Ncbps/12)*mod(k,12)+floor(k/12);  
    s=ceil(Ncbps/2);  
    jk=s*floor(mk/s)+mod(mk+Ncbps-floor(12*mk/Ncbps),s);  
    [x,int_idx]=sort(jk);  
    Ry=[+1 -1];  
    Iy=[0 0];  
    qamconst=complex(Ry,Iy);  
    qamconst=qamconst(:);  
    bitspersymbol=1;  
    CPsel=[(256-G*256+1):256 1:256];  
    CPremove=[(256*G+1):(256+G*256)];  
    coderate=1/2;  
    disp('Esquema de modulación BPSK con rata de codificación 1/2');  
    %para cada valor de SNR, los diferentes valores, definidos por el  
    %estándar
```

```

if (9.4<=cSNR&cSNR<11.2)
    inputsize=184; %tamaño de trama
    seqafterrand=inputsize+8;
    shortening=[1:32];
    shorteningRx=[1:23];
    punvec=reshape([1 0 , 1 1],4,1);%cálculo del vector de perforado,
    para una rata de 2/3
    Ncbps=384; %selector RS 48*8
    k=0:Ncbps-1;
    mk=(Ncbps/12 )*mod(k,12)+floor(k/12);
    s=ceil(Ncbps/2);
    jk=s*floor(mk/s)+mod(mk+Ncbps-floor(12*mk/Ncbps),s);
    [x,int_idx]=sort(jk);
    Ry=ones(2,1)*[+1 -1];
    Iy=([+1 -1]')*ones(1,2);
    qamconst=complex(Ry,Iy);
    qamconst=qamconst(:)/sqrt(2);
    bitspersymbol=2;
    CPsel=[ (256-G*256+1):256 1:256];
    CPremove=[ (256*G+1):(256+G*256) ];
    coderate=1/2;
    disp('Esquema de modulacion QPSK con rata de codificación 1/2');

elseif (11.2<=cSNR&cSNR<16.4)
    inputsize=280;
    seqafterrand=inputsize+8;
    shortening=[1:40];
    shorteningRx=[1:35];
    punvec=reshape([1 0 1 0 1, 1 1 0 1 0],10,1);%rata convolucional 5/6
    Ncbps=384; %RS 48*8
    k=0:Ncbps-1;
    mk=(Ncbps/12 )*mod(k,12)+floor(k/12);
    s=ceil(Ncbps/2);
    jk=s*floor(mk/s)+mod(mk+Ncbps-floor(12*mk/Ncbps),s);
    [x,int_idx]=sort(jk);
    Ry=ones(2,1)*[+1 -1];
    Iy=([+1 -1]')*ones(1,2);
    qamconst=complex(Ry,Iy);
    qamconst=qamconst(:)/sqrt(2);
    bitspersymbol=2;
    CPsel=[ (256-G*256+1):256 1:256];
    CPremove=[ (256*G+1):(256+G*256) ];
    coderate=3/4;
    disp('Esquema de modualción QPSK rata de codificación 3/4');

elseif (16.4<=cSNR&cSNR<18.2)
    inputsize=376;
    seqafterrand=inputsize+8;
    shortening=[1:64];
    shorteningRx=[1:47];
    punvec=reshape([1 0 , 1 1],4,1);%rata convolucional 2/3
    Ncbps=768; %RS 96*8
    k=0:Ncbps-1;
    mk=(Ncbps/12 )*mod(k,12)+floor(k/12);
    s=ceil(Ncbps/2);
    jk=s*floor(mk/s)+mod(mk+Ncbps-floor(12*mk/Ncbps),s);

```

```

[x,int_idx]=sort(jk);
Ry=ones(4,1)*[+1 +3 -1 -3];
Iy=([+1 +3 -3 -1])*ones(1,4);
qamconst=complex(Ry,Iy);
qamconst=qamconst(:)/sqrt(10);
bitspersymbol=4;
CPsel=[(256-G*256+1):256 1:256];
CPremove=[(256*G+1):(256+G*256)];
coderate= 1/2;
disp(' Esquema de modulación 16QAM con codificación de 1/2');

elseif (18.2<=cSNR&cSNR<22.7)
    inputsize=568;
    seqafterrand=inputsize+8;
    shortening=[1:80];
    shorteningRx=[1:71];
punvec=reshape([1 0 1 0 1, 1 1 0 1 0],10,1);%rata convolucional 5/6
Ncbps=768; %RS 96*8
k=0:Ncbps-1;
mk=(Ncbps/12 )*mod(k,12)+floor(k/12);
s=ceil(Ncbps/2);
jk=s*floor(mk/s)+mod(mk+Ncbps-floor(12*mk/Ncbps),s);
[x,int_idx]=sort(jk);
Ry=ones(4,1)*[+1 +3 -1 -3];
Iy=([+1 +3 -3 -1])*ones(1,4);
qamconst=complex(Ry,Iy);
qamconst=qamconst(:)/sqrt(10);
bitspersymbol=4;
CPsel=[(256-G*256+1):256 1:256];
CPremove=[(256*G+1):(256+G*256)];
coderate= 3/4;
disp('Esquema de modualción 16QAM rata de codificación 3/4');

elseif (22.7<=cSNR&cSNR<24.4)
    inputsize=760;
    seqafterrand=inputsize+8;
    shortening=[1:108];
    shorteningRx=[1:95];
punvec=reshape([1 0 1 , 1 1 0 ],6,1);%rata convolucional 3/4
Ncbps=1152; % RS 144*8
k=0:Ncbps-1;
mk=(Ncbps/12 )*mod(k,12)+floor(k/12);
s=ceil(Ncbps/2);
jk=s*floor(mk/s)+mod(mk+Ncbps-floor(12*mk/Ncbps),s);
[x,int_idx]=sort(jk);
Ry=ones(8,1)*[+3 +1 +5 +7 -3 -1 -5 -7 ];
Iy=([+3 +1 +5 +7 -3 -1 -5 -7 ]')*ones(1,8);
qamconst=complex(Ry,Iy);
qamconst=qamconst(:)/sqrt(42);
bitspersymbol=6;
CPsel=[(256-G*256+1):256 1:256];
CPremove=[(256*G+1):(256+G*256)];
coderate= 2/3;
disp('Esquema de modualción 64QAM rata de codificación 2/3');

elseif 24.4<=cSNR
    inputsize=856;

```

```

seqafterrand=inputsize+8;
shortening=[1:120];
shorteningRx=[1:107];
punvec=reshape([1 0 1 0 1, 1 1 0 1 0],10,1);%rata convolucional 5/6
Ncbps=1152; % RS 144*8
k=0:Ncbps-1;
mk=(Ncbps/12)*mod(k,12)+floor(k/12);
s=ceil(Ncbps/2);
jk=s*floor(mk/s)+mod(mk+Ncbps-floor(12*mk/Ncbps),s);
[x,int_idx]=sort(jk);
Ry=ones(8,1)*[+3 +1 +5 +7 -3 -1 -5 -7 ];
Iy=([+3 +1 +5 +7 -3 -1 -5 -7 ]')*ones(1,8);
qamconst=complex(Ry,Iy);
qamconst=qamconst(:)/sqrt(42);
bitspersymbol=6;
CPsel=[(256-G*256+1):256 1:256];
CPremove=[(256*G+1):(256+G*256)];
coderate= 3/4;
disp('Esquema de modualción 64QAM rata de codificación 3/4');
end

```

Aquí se activa ó desactiva el bloque AMC

```

choice=input('Teclee 0 para activar AMC, 1 para desactivarlo');
if choice==1
    Pulse=2;
    delayswitch=2;
    delayBER=2*inputsize;
else
    Pulse=1;
    delayswitch=0;
    delayBER=0;
end

```