LAMPIRAN-LAMPIRAN

A

PERANGKAT LUNAK

% Program Utama

```
% Bilangan prima yang dipilih (sama dengan w / bobot Hamming
% untuk kode prima)
p=3;
%Pembangkitan data
data=[1 1 0 1 1];
%Pengulangan data agar sama panjang dengan kode prima yang
% digunakan
data_ulang=ulang(p,data);
% Pembangkitan prime sequence
brs=baris_prima(p);
% Pembangkitan kode prima
kode=kode_prima(brs);
% Hitung korelasi silang
kode_used1=kode(1,:);
kode_used2=kode(2,:);
 % Batas hitung korelasi silang
```

```
% Hitung autokorelasi
 kode_used1=kode(4,:);
 kode_used2=kode_used1;
% Batas hitung autokorelasi
for k=1:length(data)
    data_termod_kali1((k-1)*p.^2+1:k*p.^2)=...
        kali(data_ulang((k-1)*p.^2+1:k*p.^2),kode_used1);
    data_termod_kali2((k-1)*p.^2+1:k*p.^2)=...
        kali(data_ulang((k-1)*p.^2+1:k*p.^2),kode_used2);
end;
for m=1:length(data)
    bantu1(1:p.^2)=data_termod_kali1((m-1)*p.^2+1:m*p.^2);
    bantu2(1:p.^2)=data_termod_kali2((m-1)*p.^2+1:m*p.^2);
%     bantu2=bantu1;
    for k=1:p.^2
        kor(k)=hitung_korelasi(bantu1,circshift(bantu2,[0 k-1]));
    end;
    korelasi((m-1)*p.^2+1:m*p.^2)=kor;
    clear kor;
end;
figure;stem(1:length(korelasi),korelasi);grid;
title('Perhitungan korelasi silang untuk C1 dan C2 pada GF(7)');
% title('Perhitungan autokorelasi untuk C4 pada GF(7)');
% Untuk menampilkan sinyal yang dikirimkan dengan termodulasi ASK
for m=1:length(data_termod_kali1)
    sinyal_tx((m-1).*20+1:m.*20)=sinyal(data_termod_kali1(m));
end;
```

```
sinyal_tx_noise=sinyal_tx+0.01.*randn(1,length(sinyal_tx));
figure;plot(sinyal_tx);grid;
title('Sinyal yang dikirimkan untuk C1 dan C2 pada GF(3)');
figure;plot(sinyal_tx_noise);grid;
title('Sinyal yang dikirimkan plus noise untuk C3 dan C4 pada GF(7)');
% Perhitungan BER jika ada lebih dari p user
% (BER sebagai fungsi jumlah user dalam sistem), kanal AWGN, single path
% Inisialisasi kanal
SNR = 3;
T = 50000;
alpha = 1;
SNR_dec = 10.^(SNR / 10);
ku = 1;
k=1;
```

```
N = p.^2; %  the length of spread code
Matcodes=[kode;kode_multi_prima(p)]
% Mulai perhitungan dengan MUD : matched filter
for K=1:1:size(Matcodes,1)  %K jumlah user
   A=floor((0:(K-1))*5/K)+1;
   Aavg = sum(A.^2,2)/T;%
   C = Matcodes(K,:);
   Eb = C(:,ku)' * C(:,ku)*Aavg(ku);
   mf = C(:,ku);
   sigma = Eb/SNR_dec(k)/2;
   data=randuni(1000);
   Sentbit(K,:)=ulang(p,data);
   kode_used=Matcodes(K,:);
   kode_used=circshift(kode_used,[0 K-1]);
   for m=1:length(data)
      Y(K,(m-1)*p.^2+1:m*p.^2)=...
         kali(Sentbit(K,(m-1)*p.^2+1:m*p.^2),kode_used)...
         +0.01.*randn(1,p.^2);
   end;
   for m=1:length(data)
      Receivedbit1(K,(m-1)*p.^2+1:m*p.^2)=sign(mf.*Y(K,(m-1)*p.^2+1:m*p.^2));
   end;
   Perr1(K) = sum(abs(Receivedbit1(K,:)-Sentbit(K,:)))/2/T;

   % Perhitungan deteksi (MUD) : decorrelator
   for pp=1:length(C)
      if C(pp)==0
```

```
            C(pp)=C(pp)+10*eps;
        end;
    end;
    Receivedbit2 = sign(inv(C'*C)*C'*Y(K,:));
    Perr2(K) = sum(abs(Receivedbit2(K,:)-Sentbit(K,:)))/2/T;
end;
% kode prima
%Pengulangan data agar sama panjang dengan kode prima yang
% digunakan
data_ulang1=ulang(p,data);
kode_sink=brs_prima_sink(p);
b=cell2mat(kode_sink);
x=zeros(p,p.^2);
row_prime_sync=zeros(p.^2,p);
for k=1:p
    x(k,:)=b(1,:,k);
end;
            row=1;
            kol=1;
for m=1:p
    row_prime_sync((m-1).*p+1:m.*p,:)=x((row-1).*p+1:row.*p,(kol-1).*p+1:kol.*p);
    kol=kol+1;
end;

kode_prima_sink=kode_prima_sinkron(row_prime_sync);
kode1=kode_prima_sink;
% Pemilihan kode prima untuk user tertentu
```

```
% indeks_kode=randperm(p);
% kode_used=kode(indeks_kode(1),:);

% Hitung korelasi silang
kode_used11=kode1(1,:);
kode_used21=kode1(4,:);
% Batas hitung korelasi silang

% Hitung autokorelasi
% kode_used1=kode(4,:) % untuk contoh dari buku pake C3 pada GF(7)
% kode_used2=kode_used1;
% Batas hitung autokorelasi
for k=1:length(data)
   data_termod_kali11((k-1)*p.^2+1:k*p.^2)=...
      kali(data_ulang1((k-1)*p.^2+1:k*p.^2),kode_used11);
   data_termod_kali21((k-1)*p.^2+1:k*p.^2)=...
      kali(data_ulang1((k-1)*p.^2+1:k*p.^2),kode_used21);
end;



for m=1:length(data)
   bantu11(1:p.^2)=data_termod_kali11((m-1)*p.^2+1:m*p.^2);
   bantu21(1:p.^2)=data_termod_kali21((m-1)*p.^2+1:m*p.^2);
%     bantu2=bantu1;    % Ini di-uncomment untuk hitung autokorelasi
   for k=1:p.^2
      kor1(k)=hitung_korelasi(bantu11,circshift(bantu21,[0 k-1]));
```

```
    end;
    korelasi1((m-1)*p.^2+1:m*p.^2)=kor1;
    clear kor1;
end;

figure;stem(1:length(korelasi1),korelasi1);grid;
title('Perhitungan korelasi silang untuk C1 dan C4 pada GF(3)');
% title('Perhitungan autokorelasi untuk C4 pada GF(7)');

% Untuk menampilkan sinyal yang dikirimkan dengan termodulasi ASK
for m=1:length(data_termod_kali11)
    sinyal_tx1((m-1).*20+1:m.*20)=sinyal(data_termod_kali11(m));
end;

sinyal_tx_noise1=sinyal_tx1+0.01.*randn(1,length(sinyal_tx1));

figure;plot(sinyal_tx1);grid;
title('Sinyal yang dikirimkan untuk C1 dan C4 pada GF(7)');
```

```
figure;plot(sinyal_tx_noise1);grid;
title('Sinyal yang dikirimkan plus noise untuk C3 dan C4 pada GF(7)');

% Perhitungan BER jika ada lebih dari p user
% (BER sebagai fungsi jumlah user dalam sistem), kanal AWGN, single path
N1 = 2*p.^2-p; %  the length of spread code
Matcodes1=[kode1;kode_multi_sinkron(p)]
k=1;
% Mulai perhitungan dengan MUD : matched filter
for K=1:1:size(Matcodes1,1)  %K jumlah user
   A=floor((0:(K-1))*5/K)+1;
   Aavg = sum(A.^2,2)/T;%
   C = Matcodes1(K,:);
   Eb = C(:,ku)' * C(:,ku)*Aavg(ku);
   mf = C(:,ku);
   sigma = Eb/SNR_dec(k)/2;
   data=randuni(1000);
   Sentbit1(K,:)=ulang(p,data);
   kode_used1=Matcodes1(K,:);
   kode_used1=circshift(kode_used1,[0 (p.^2)-1]);
   for m=1:length(data)
      Y1(K,(m-1).*(p.^2)+1:m.*(p.^2))=...
         kali(Sentbit1(K,(m-1).*(p.^2)+1:m.*(p.^2)),kode_used1)+...
         0.01.*randn(1,(p.^2));
   end;
```

```
    for pp=1:length(C)
        if C(pp)<=0.05
            C(pp)=C(pp)+10*eps;
        end;
    end;

    for m=1:length(data)
        Receivedbit11(K,:)=sign(mf.*Y1(K,:));
    end;
    Perr11(K) = sum(abs(Receivedbit11(K,:)-Sentbit1(K,:)))/2/T;

    Receivedbit21 = sign(inv(C'*C)*C'*Y1(K,:));
end;
sent_aja=Sentbit1(1:size(Receivedbit21,1),:);

for jj=1:size(Receivedbit21,1)
    Perr21(jj) = sum(abs(Receivedbit21(jj,:)-sent_aja(jj,:)))/2/T;
end;
figure;
semilogy(1:1:size(Matcodes,1), Perr2(1:1:size(Matcodes,1)), 'ro-');hold on;grid;
xlabel('Jumlah user');ylabel('BER');title('MUD dengan decorrelator');
semilogy(1:1:size(sent_aja,1), Perr21(1:1:size(sent_aja,1)), 'b-');hold on;grid;
```

```
% Pengecekan apakah ada user menggunakan kode yang sama
help1=0;
for dd=1:size(Matcodes,1)
   for ee=1:size(Matcodes,1)
      if dd~=ee
         if Matcodes(ee,:)==Matcodes(dd,:)
            help1=help1+1;
         end;
      end;
   end;
end;
help1
```

Function Baris prima

```
function keluar=baris_prima(masuk)
% Function ini untuk menghasilkan barisan prima (prime sequence)
% Catatan : masukan harus berupa bilangan prima
% Variabel masukan : masuk = bilangan prima
% Variabel keluaran : keluar = barisan prima
cek=isprime(masuk);
if cek == 0
   error('Bilangan yang dimasukkan BUKAN bilangan prima');
end;
p = masuk;  % bilangan prima yang diinputkan

for m = 1 : p
   for n=1:p
      S(m,n) = mod ((m-1).* (n-1),p);
   end;
end;
keluar=S;
```

Function kode prima

```
function keluar=kode_prima(masuk)
% Function ini untuk menghasilkan kode prima (prime code)
% Catatan : masukan berupa barisan prima
% Variabel masukan : masuk
% Variabel keluaran : keluar
keluar=zeros(size(masuk,1),size(masuk,1).*size(masuk,2));

 for m=1:size(masuk,1)
    temp=masuk(m,:);
    for n=1:length(temp)
        keluar(m,(n-1).*size(masuk,2)+temp(n)+1)=1;
    end;
 end;
```

Function kode prima sinkron

```
function keluar=kode_prima_sinkron(masuk)
% Function ini untuk menghasilkan kode prima (prime code)
% yang sudah disinkronisasi
% Catatan : masukan berupa barisan prima
% Variabel masukan : masuk
% Variabel keluaran : keluar
keluar=zeros(size(masuk,2).*size(masuk,2),size(masuk,2).*size(masuk,2));

 for m=1:size(masuk,1)
    temp=masuk(m,:);
    for n=1:length(temp)
        keluar(m,(n-1).*size(masuk,2)+temp(n)+1)=1;
    end;
 end;
```

Function randuni

```
function[p]=randuni(N)
% Function ini membangkitkan data biner
% unipolar secara random
% Variabel masukan : N = bilangan bulat (>=2) random
% Variabel keluaran : p = data random unipolar

for i=1:N
   temp=rand;
   if (temp<0.5)
      data(1,i)=0;
   else
      data(1,i)=1;
   end;
end;
p=data(1,:);
```

Function Sinyal

```
function keluar=sinyal(masuk)
% Function ini untuk modulasi ASK
% Variabel masukan : masuk
% Variabel keluaran : keluar
t=0.05:.05:1;
f=1;
A=1;

if masuk==1
   keluar=A.*sin(2.*pi.*f.*t);
else
   keluar=zeros(1,length(t));
end;
```

Function hitung korelasi

```
function [keluar]=hitung_korelasi(masuk1,masuk2)
% Function untuk menghitung nilai korelasi antara
% dua buah kode (korelasi sendiri atau korelasi silang)
% Variabel masukan : masuk1=kode pertama
%                    masuk2=kode kedua
% Variabel keluaran : keluar = nilai korelasi
if (length(masuk1) ~= length (masuk2))
   error('Panjang kode tidak sama');
end;

keluar=0;
for k=1:length(masuk1)
   if (masuk1(k)==1 & masuk2(k)==1)
      keluar=keluar+1;
   end;
end;
```

Function ulang

```
function keluar=ulang(p,masuk)
% Function ini untuk mengulang data user
% sesuai dengan periode chip yang digunakan(p^2)
% Variabel masukan : p =bilangan prima
%                    masuk = data user
% Variabel keluaran : keluar = data user yang
%                     sudah di-repetisi (diulang)
%                     sehingga panjangnya sama
%                     dengan banyak data * p^2

for m=1:length(masuk)
   keluar((m-1)*p^2+1:m*p^2)=masuk(m);
end;
```

Function kali

```
function keluar = kali(masuk1,masuk2)
% Function ini untuk operasi perkalian
% Variabel masukan : masuk1 dan masuk2
% Variabel keluaran : keluar
if length(masuk1) ~= length(masuk2)
   error('Dimensi kedua sinyal masukan berbeda');
end;

for k=1:length(masuk1)
   if masuk1(k)==0 | masuk2(k)==0
      keluar(k)=0;
   else
      keluar(k)=1;
   end;
end;
```

Function kode multi prima

```
function [keluar]=kode_multi_prima(masuk)
% Function ini untuk menambahkan kode
% untuk user lain selain kode prima yang ada
% Variabel masukan : masuk = bilangan prima
% Variabel keluaran : keluar = kode prima tambahan
for m=1:masuk.^2-masuk
   x=zeros(1,masuk.^2);
   cek=randperm(masuk.^2);
   for k=1:masuk
      x(cek(k))=1;
   end;
   tes(masuk+m,:)=x;
   clear x;
end;
keluar=tes(masuk+1:end,:);
```