BAB 3

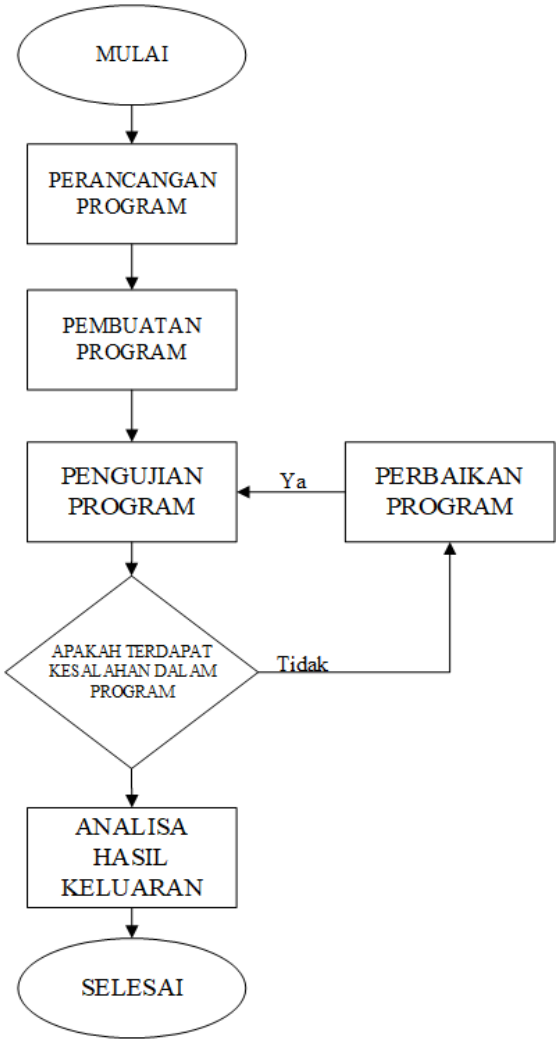
METODOLOGI PENELITIAN

3.1. Alat Yang Digunakan

Penelitian ini bersifat simulasi, untuk dapat melihat unjuk kerja FBMC dengan menggunakan modulasi OQAM pada sistem komunikasi SISO. Alat yang digunakan pada penelitian ini adalah *software* MATLAB 2018b.

3.2.Alur Penelitian

Alur penelitian ini dapat ditunjukkan pada Gambar berikut ini:



Gambar 3.1 *Flowchart* Rangkaian Simulasi Program

3.3.Parameter Simulasi

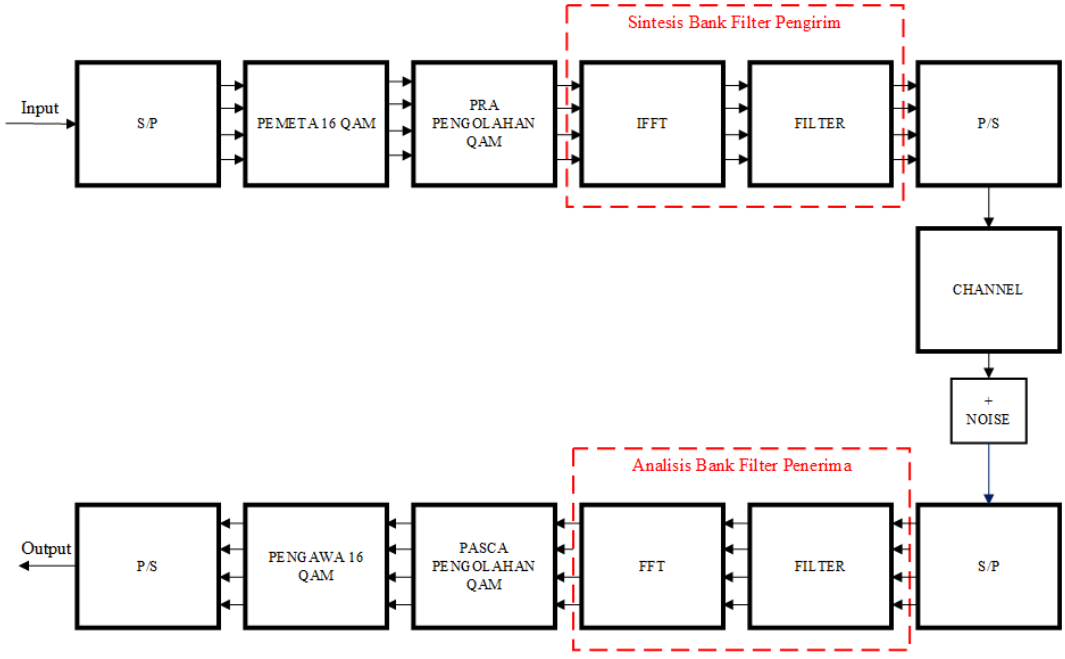
Parameter simulasi yang digunakan dalam simulasi, dapat dilihat pada Tabel 3.1 berikut ini:

Tabel 3.1 Parameter Simulasi

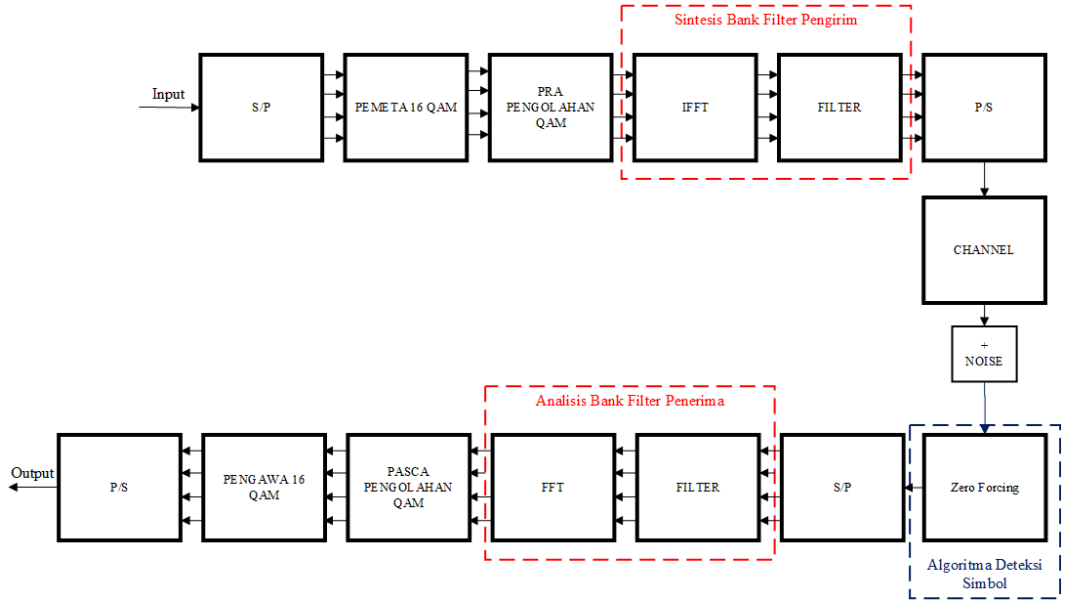
Simbol	Parameter	Nilai
Ntx	Jumlah antena pengirim	1
Nrx	Jumlah antena penerima	1
M	Modulasi dasar	16 QAM
Ml	Jumlah level modulasi	4

3.4.Pemodelan Sistem SISO FBMC OQAM

Pemodelan sistem SISO FBMC OQAM dapat dijelaskan pada gambar berikut ini:



Gambar 3.2 SISO FBMC dari sisi Pengirim dan Penerima



Gambar 3.3 SISO FBMC dari sisi Penerima dan Pengirim dengan *Zero Forcing*

3.4.1. Transmisi Citra *QR Code* Pada *Transmitter*

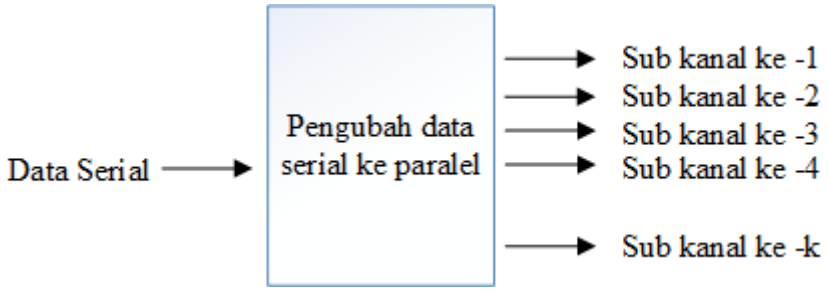
3.4.1.1.Data Masukan

Data masukan dari simulasi *Filter Bank Multicarrier* OQAM ini berupa citra *QR code* dalam format *file* .png. Data masukan berupa citra *QR code* tersebut memiliki dimensi 80 x 80 *pixels*. Adapun kode program dari data masukan adalah sebagai berikut:

```
%% input QR Code
data_input =[];
bar = imread('nbar1.png');
gbar = rgb2gray(bar);
inputbar = imbinarize(gbar);
data_input = [data_input,inputbar(:)]';
```

3.4.1.2.Pengubah Data Serial ke Paralel

Pada bagan ini berfungsi merubah bit serial ke paralel serta mengelompokkan bit sesuai dengan level modulasi yang digunakan. Seperti yang ditunjukkan pada Gambar 3.4 berikut ini:



Gambar 3.4 Pengubah data seri ke paralel

Adapun kode proram yang digunakan untuk pengubah serial ke paralel adalah:

```
sp_tx = reshape(data_input, ml, length(data_input)/ml);
sp_tx = sp_tx';
```

Data masukan dari citra yang sudah dirubah dalam bentuk biner kemudian dirubah menjadi bentuk paralel berdasarkan level modulasi yang digunakan. Dalam penelitian ini menggunakan level modulasi 16 QAM atau dibagi menjadi 4 bit–4 bit.

3.4.1.3.Pemeta 16 QAM

Hasil keluaran dari pengubah data serial ke paralel selanjutnya di petakan dengan modulasi 16 QAM. Tujuannya untuk mengubah data masukan biner menjadi bilangan kompleks $S_k = I_k + jQ_k$, dengan k adalah variabel simbol, I adalah *inphase* atau bilangan riil, dan Q adalah bilangan imajiner. Modulasi 16 QAM artinya setiap simbol yang diterima pada pemetaan ini terdiri dari empat bit. Berdasarkan penjelasan tersebut, setiap simbol yang dikirimkan dari S/P dipetakan dengan kode program berikut ini:

```
for c=1:(length(sp_tx))
    qammod(c,:)=(1/sQRt(10))*((1-2*(sp_tx(c,4)))*(2-(1-
2*(sp_tx(c,2))))+sQRt(-1)*(1-2*(sp_tx(c,3)))*(2-(1-
2*(sp_tx(c,1)))));
end
end
```

“sp_tx” merupakan variabel hasil keluaran dari proses sebelumnya yaitu serial ke paralel , “qammod” merupakan variabel penampung pemeta 16 QAM,

“c=1:(length(sp_tx))” merupakan proses perulangan pemeta 16 QAM sesuai banyaknya Panjang sp_tx. Dari proses ini dapat dihasilkan bentuk bilangan kompleks untuk pemeta 16 QAM seperti pada tabel berikut ini:

Tabel 3.2 Pemeta 16 QAM

No	Biner 4 bit				Bilangan Kompleks
	Bit -4	Bit-3	Bit-2	Bit-1	
1	0	0	0	0	0,3162 + 0,3162*i*
2	0	0	0	1	-0,3162 + 0,3162*i*
3	0	0	1	0	0,3162 - 0,3162*i*
4	0	0	1	1	-0,3162 - 0,3162*i*
5	0	1	0	0	0,9486 + 0,3162*i*
6	0	1	0	1	-0,9486 + 0,3162*i*
7	0	1	1	0	0,9486 - 0,3162*i*
8	0	1	1	1	-0,9486 - 0,3162*i*
9	1	0	0	0	0,3162 + 0,9486*i*
10	1	0	0	1	-0,3162 + 0,9486*i*
11	1	0	1	0	0,3162 - 0,9486*i*
12	1	0	1	1	-0,3162 - 0,9486*i*
13	1	1	0	0	0,9486 + 0,9486*i*
14	1	1	0	1	-0,9486 + 0,9486*i*
15	1	1	1	0	0,9486 - 0,9486*i*
16	1	1	1	1	-0,9486 - 0,9486*i*

3.4.1.4. Pra Pengolahan OQAM

Pada proses pra pengolahan OQAM, simbol yang dikirimkan dipecah menjadi 2 yaitu simbol ganjil dan genap. Simbol ganjil genap didapat dari urutan simbol yang dikirimkan. Simbol tersebut diolah di dalam pra pengolahan OQAM yang terdiri dari 2 operasi yaitu:

1. Proses pengubahan bilangan kompleks menjadi bilangan riil.
2. Proses perkalian dengan $\theta_{k,n}$.

Pada operasi pertama simbol ganjil genap yang terdiri dari bilangan riil dan imajiner masing – masing akan dipisahkan. Lalu masing-masing bilangan riil dan

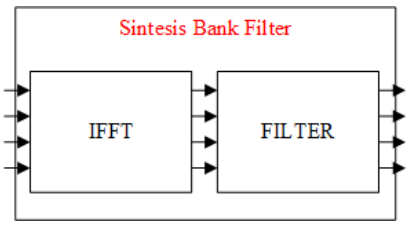
imajiner mengalami peningkatan jumlah (*upsampling* sebesar 2 kali) yang diikuti dengan pergeseran setengah fasa sebesar 90◦ (1 fasa = 180◦). Adapun kode program yang digunakan adalah:

```
for n = 1 : length(qammod_atas)
    if (rem(n, 2) == 1) %odd
        oqam_pre(n,1)=imag(qammod_atas(n))*((j).^(n));
        oqam_pre(n,2)=real(qammod_atas(n))*((j).^(n));
    else %even
        oqam_pre(n,1)=real(qammod_atas(n))*((j).^(n));
        oqam_pre(n,2)=imag(qammod_atas(n))*((j).^(n));
%dikolom kedua karena ada delay
    end
end
```

Perbedaan di dalam proses pra pengolahan terletak pada tahap pergerseran fasa. Simbol genap terjadi pegeseran fasa pada bilangan imajiner. Sedangkan simbol ganjil terjadi pergesaran fasa pada bilangan riil. Pada proses kedua simbol yang dihasilkan oleh proses pertama dikalikan dengan θ_{kn} untuk mendapat nilai riilnya.

3.4.1.5. Sintesis Bank Filter

Pada proses sintesis bank filter ini terdiri dari yaitu kebalikan transformasi *Fourier* atau IFFT serta proses filter.



Gambar 3.5 Proses Sintesis Bank Filter

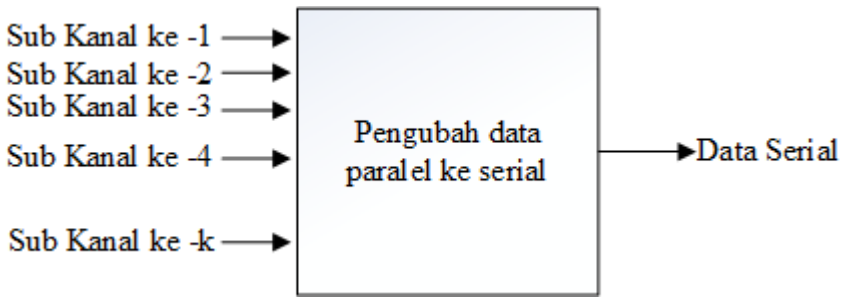
Filter pada simulasi ini diasumsikan menggunakan filter ideal. Filter ini di asumsikan bahwa *channel* sama dengan satu artinya bahwa berapapun data yang masuk akan ditransmisikan dengan pengalinya sama dengan satu. Sehingga simbol yang akan ditransmisikan langsung diteruskan tanpa adanya data yang dihilangkan.

```
function[ppn_sfb]=synthesis(oqam_pre)
%Transformation Block
ifft_out = ifft(oqam_pre);
%PPN Filter
z=1; %contoh filter = 1
for x=1:length(ifft_out)
    ppn_sfb(x,:)=ifft_out(x)*z;
end
% ppn_sfb=npr_analysis(c,ifft_out);
end
```

"oqam_pre" merupakan variabel baru yang menjelaskan simbol baru keluaran pra pengolahan OQAM. "ifft_out" merupakan variabel baru yang menjelaskan tentang simbol baru keluaran pra pengolahan oqam yang dikalikan dengan kebalikan transformasi *Fourier*. Kemudian setiap simbol keluaran ifft disimbolkan dengan "ppn_sfb" yang selanjutnya dikenakan dengan filter. Filter dalam penelitian ini diasumsikan menggunakan filter ideal, yang berarti tidak ada simbol yang dihilangkan.

3.4.1.6. Pengubah Data Paralel ke Serial

Pengubah paralel ke seri di sisi pengirim, berfungsi untuk mengubah keluaran dari sintesis bank filter yang semula bentuknya data paralel menjadi bentuk seri, untuk dapat dikirimkan menjadi 1 baris.



Gambar 3.6 Pengubah Data Paralel ke Serial

Adapun kode program yang digunakan untuk pengubah paralel ke serial adalah sebagai berikut :

```
ps_tx = reshape(ppn_sfb,1,[]);
```

3.4.1.7. Pemodelan Kanal Transmisi

Kanal transmisi yang digunakan pada simulasi ini adalah kanal AWGN. Pada kanal AWGN diasumsikan memiliki derau yang terdistribusi normal (Gaussian). Derau AWGN terdistribusi normal dengan nilai rata-rata adalah nol. Derau ini bersifat acak dan bersifat menambahkan sinyal asli seperti yang telah dijelaskan pada Persamaan 2.5. Kode program yang digunakan untuk pemodelan kanal adalah sebagai berikut:

```
rx = (h .* source1) + noise/100;
```

Notasi "h" merupakan kanal AWGN yang memiliki Panjang yang telah disamakan dengan data masukan serta derau yang diperoleh secara *random*. "source1" merupakan variabel simbol yang dikirimkan dari sisi pengirim.

3.4.2. Transmisi Citra *QR Code* Pada *Receiver*

3.4.2.1. Deteksi Simbol *Zero Forcing*

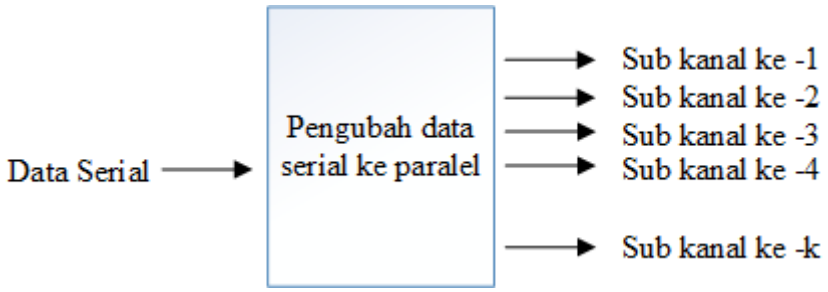
Proses penerimaan diawali dengan mendeteksi simbol yang dikirmkan. Proses ini bertujuan untuk mendapatkan sinyal asli yang dikirimkan secara digital. Algoritma deteksi simbol yang digunakan adalah *Zero Forcing*, dimana algoritma deteksi simbol ini bekerja dengan cara menginversikan kanal **H**, dikalikan dengan sinyal yang dikirimkan dan ditambahkan derau yang AWGN yang diperoleh secara *random* sebelumya. Deteksi simbol *Zero Forcing* didapatkan menggunakan kode program berikut ini:

```
rx2 = (1./h).*rx; %Zero Forcing Equalizer
```

Pada penjelasan sebelumnya, "rx" merupakan variabel penampung data setelah ditambahkan derau *random*. Maka untuk "rx2" merupakan variabel yang menampung simbol setelah dikenakan atau melewati deteksi simbol. Pada penelitian ini jenis kanal yang digunakan adalah kanal AWGN *(Additive White Gaussian Noise)* dimana jenis kanal ini bervariasi berubah berdasarkan derau yang ditambahkan. Sedangkan pada sisi ekualisasi *zero forcing* menggunakan ekualisasi yang tetap.

3.4.2.2. Pengubah Data Serial ke Paralel

Pengubah data seri menjadi paralel di sisi penerima, berfungsi untuk mengubah keluaran dari antena penerima bentuk paralel. Untuk dapat dilanjutkan ke proses selanjutnya yaitu proses analisis bank filter.



Gambar 3.7 Pengubah Data Serial Ke Paralel

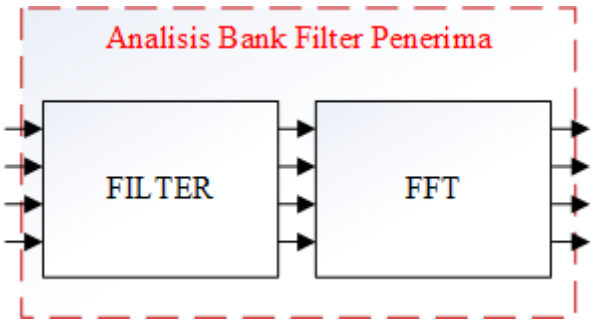
Adapun kode program yang digunakan adalah:

```
sp_rx2 = reshape(rx2,[],1);
```

"sp_rx2" adalah variabel yang menampung hasil dari pengubah serial ke paralel. "reshape(rx2,[],1);" digunakan untuk mengubah data pada "rx2" dari banyak baris menjadi 1 kolom.

3.4.2.3. Analisis Bank Filter

Pada proses analisis bank filter ini merupakan proses kebalikan dari sintesis bank filter. Proses ini terdiri dari proses yaitu proses transformasi *Fourier* serta proses filter. Dapat ditunjukkan pada Gambar 3.8 berikut ini:



Gambar 3.8 Proses Analisis Bank Filter

Kode program yang digunakan untuk proses analisis bank filter adalah sebagai berikut:

```
function[ppn_afb_out]=analysis(a,b,sp_rx)
%Transformation Block
```

```
fft_out = fft(sp_rx);
% %PPN Filter
z=1; %contoh filter = 1
for x=1:length(fft_out)
    ppn_afb(:,x)=fft_out(x)*z;
end
ppn_afb_out=reshape(ppn_afb,a,b);
end
```

Pada proses analisis bank filter ini, masing – masing simbol akan dikenakan dengan filter. Filter yang digunakan masih sama dengan sebelumnya yaitu filter ideal, artinya simbol akan langsung diteruskan ke proses selanjutnya tanpa ada simbol yang dihilangkan. “ppn_afb” merupakan variabel keluaran filter yang kemudian dikalikan dengan transformasi *Fourier*. Hasil dari proses perkalian akan ditampung pada variabel “ppn_afb_out”.

3.4.2.4. Pasca Pengolahan OQAM

Pasca pengolahan OQAM merupakan kebalikan dari bagan pra pengolahan OQAM. Blok pasca pengolahan seperti yang ditunjukkan pada Gambar 2.9, yang memiliki dua struktur yang berbeda berdasarkan urutan kanal ganjil atau genap. Kode program dari pasca pengolahan OQAM adalah sebagai berikut:

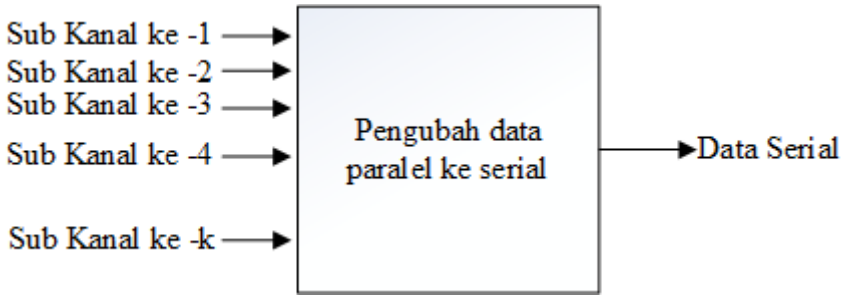
```
for n = 1 : length(ppn_afb_out)
    if (rem(n, 2) == 1) %odd
        oqam_post=ppn_afb_out*((-1*j).^(n));
        oqam_post=real(oqam_post);
   oqam_post_atas(n,1)=oqam_post(n,2)+j*oqam_post(n,1);
    else %even
        oqam_post=ppn_afb_out*((-1*j).^(n));
        oqam_post=real(oqam_post);
oqam_post_atas(n,1)=oqam_post(n,1)+j*oqam_post(n,2);
    end
end
```

3.4.2.5. Pengawa Peta 16 QAM

Pada proses ini merupakan proses kebalikan dari pemeta 16 QAM. Proses ini setiap simbol akan dipetakan kembali sesuai dengan level modulasinya. Tujuannya untuk menentukan simbol mana yang dikirimkan oleh pengirim. Proses ini diperlukan karena pada kanal terdapat derau yang berpengaruh terhadap data yang dikirimkan dari keluaran pemeta 16 QAM. Hasil pemeta ini berupa bilangan kompleks lagi atau bilangan riil dan imajiner yang kemudian akan dikonversikan menjadi bentuk biner kembali.

3.4.2.6. Pengubah Paralel ke Serial

Blok pengubah paralel menjadi seri ditunjukkan pada Gambar 3.9. Fungsinya adalah untuk mengubah bentuk dari yang semula bentuk bit paralel menjadi bit seri, dengan mengelompokkan empat bit – empat bit biner keluaran dari pengawa peta 16 QAM, menjadi satu baris sebagai keluaran dari proses pengawa peta 16-QAM ke data serial bit.



Gambar 3.9 Pengubah data Paralel ke Serial

3.4.2.7. Data Keluaran

Di sisi keluaran, bit-bit biner tersebut dikembalikan kembali ke bentuk semula yaitu gambar (citra). Hasil bit - bit biner keluaran sinyal tersebut, dapat digunakan untuk membandingkan bit keluaran dengan bit masukan. Karena data masukan yang dimasukkan berupa citra *QR code* maka pada data keluaran juga harus berbentuk citra seperti semula. Sehingga pada data keluaran diperlukan rekonstruksi gambar agar dapat terbentuk seperti data masukan. Adapun kode program yang digunakan adalah sebagai berikut:

```
%% Coba Rekonstruksi QRcode
% Rekonstruksi hasil tanpa Zero Forcing
data_output_coba = data_output;
img_output = reshape(data_output_coba,[80,80]);
img_output_final = cast(img_output,'logical');
% Rekonstruksi hasil Zero Forcing
data_output2_coba = data_output2;
img_output2 = reshape(data_output2_coba,[80,80]);
img_output_final2 = cast(img_output2,'logical');
```