BAB III

ANALISIS DAN PERANCANGAN SISTEM

III.1. Analisis Sistem

Analisis sistem dapat didefinisikan sebagai penguraian suatu sistem yang utuh kedalam bagian-bagian komponennya dengan maksud untuk mengidentifikasi dan mengevaluasi permasalahan-permasalahan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikannya.

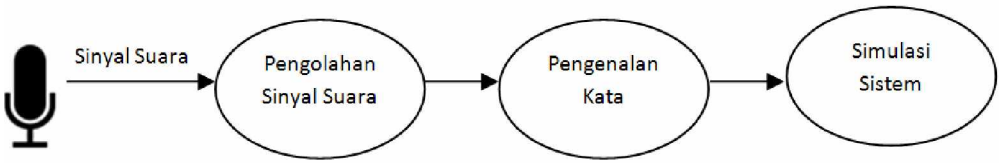
Dalam proses pembuatan suatu sistem mutlak dilakukan penelitian dan penganalisaan tentang sistem yang akan dibangun, berikut adalah beberapa analisis yang dilakukan untuk membangun Aplikasi Perintah Suara dengan Metode *Divide and Conquer* pada Simulasi Rumah Pintar.

III.1.1. Analisis Masalah

Proses pengenalan kata dan simulasi pada sistem harus melalui beberapa tahap proses agar kata dapat dikenali oleh sistem. Suara yang telah direkam berupa sinyal analog diolah menjadi sinyal digital untuk diubah menjadi data diskrit. Data suara hasil rekaman kembali diproses dalam pengolahan sinyal suara, di ekstraksi ciri fiturnya untuk mengetahui perbedaan dan ciri-ciri tiap suara agar memudahkan saat proses pengenalan kata. Setelah itu barulah sistem pengenalan kata akan mengenali ucapan dari pengguna setelah sistem memiliki *template* fitur yang akan dijadikan perbandingan. Hasil pengenalan kata dapat diproses lebih lanjut atau di implementasikan ke berbagai aplikasi.

III.1.1.1. Deskripsi Umum Sistem

Deskripsi secara umum dalam sistem perintah suara ini adalah dimulai dari tahap pengolahan sinyal suara. Tahap pengolahan sinyal suara adalah proses pengkondisian sinyal suara dimulai dari perekaman suara dan tahap *analog to digital conversion*. Setelah itu masuk pada tahap *feature extraction* dengan sebelumnya sinyal diolah menggunakan analisis *Fast Fourier Transform (FFT)*.

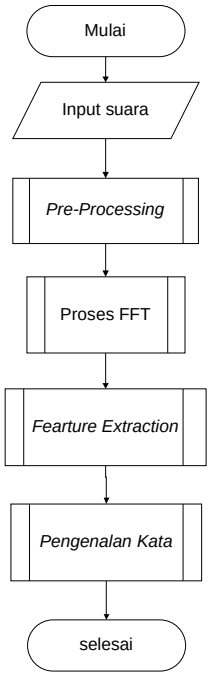


Gambar III.1. Proses Sistem Perintah Suara dan Simulasi

Gambar III.1. menunjukan garis besar proses perintah suara dan simulasi pada yang akan dirancang. Analisis FFT adalah tahap merubah sinyal suara dalam domain waktu menjadi sinyal suara dalam domain frekuensi. Tahap pengenalan kata adalah tahap yang menggunakan metode *Divide and Conquer*, tahap ini melakukan pembagian data masukan dan pencocokan pola diproses satu per satu berdasarkan urutan, lalu hasilnya digabungkan kembali dan menjadikan suatu solusi. Dalam proses pencocokan pola data baru dengan data yang ada pada *template*, dilakukan dengan fungsi *binary search* dengan teknik implementasi dari metode *Divide and Conquer*. Pada bagian simulasi sistem, hasil dari proses pengenalan kata berdasarkan *template* dimana nilai peramalan *error* terkecil saat pencocokan pola, maka itu akan menjadi keluaran atau kata yang dikenali.

III.1.2. Analisis Pengolahan Sinyal Suara

Dalam analisis pengolahan sinyal suara ini dibagi menjadi 3 tahap, tahap pertama adalah *pre-processing*, tahap kedua adalah tahap memproses sinyal berdasarkan frekuensi dengan FFT, lalu data hasil dari proses FFT diambil fiturnya pada proses *feature extraction*. Secara umum proses pengolahan sinyal suara pada program ini mengikuti alur sesuai yang ditunjukan pada gambar III.2.

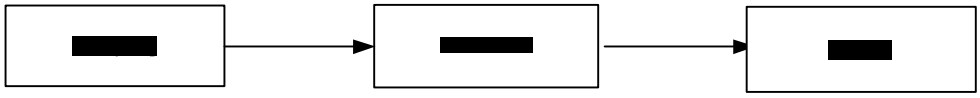


Gambar III.2. Alur Pengolahan Sinyal Suara

III.1.2.1. *Pre-Processing*

Sinyal suara yang akan diproses bersifat analog sehingga jika dilakukan pengolahan secara digital, sinyal suara tersebut harus dikonversi menjadi sinyal digital, berupa urutan angka dengan tingkat presisi tertentu yang dinamakan

analog to digital conversion dengan menggunakan *analog-to-digotal conventer* (ADC).



Gambar III.3. Konsep Kerja ADC (Analog to Digital Converter)

1. Sampling

Data berupa sinyal suara diperoleh dengan cara merekam suara melalui mikrofon yang dihubungkan dengan komputer. Perekaman suara di dalam aplikasi menggunakan bantuan *function* dari program dan aplikasi *accessories windows*, yaitu *sound recorder* dengan format *.wav* dan frekuensi sampling standar terbaik pada PC adalah 8000Hz, 11025Hz, 22050Hz dan 44100Hz. Suara dengan format *.wav* ini bisa menggunakan 8 atau 16 *bits*/*sample* dan 1 untuk *channel mono* atau 2 untuk *channel stereo*. Durasi suara yang direkam apabila lebih pendek lebih mudah untuk diambil perbedaan fiturnya. Dalam analisis ini digunakan contoh durasi rekaman yang diambil adalah 0,8 detik.

$$X = FS\ x\ dt(detik)\ x\left(\frac{bit}{8}\right) x\ j \qquad (3.1)$$

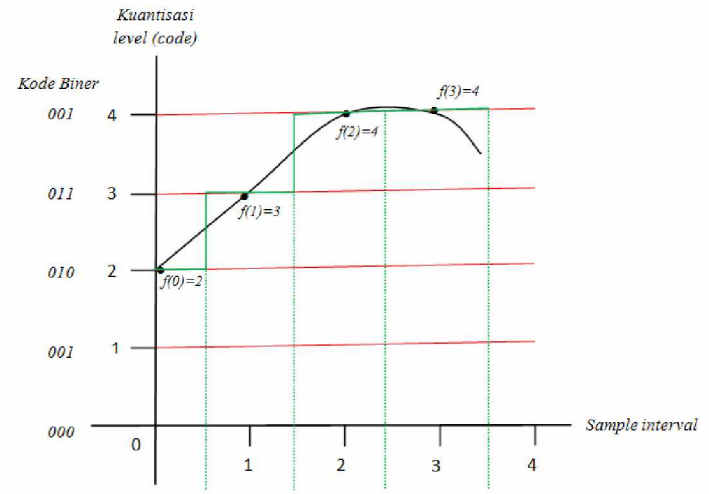
Perhitungan pada proses akuisi data untuk pengambilan sampling :

Tabel III.1. Perhitungan Sampling Data Sinyal

Frekuensi (Hz)	***Bits/ Sample***	***Channel***	**Perhitungan**	**Data Sample**
8000	8	mono	800*0.8*(8/8)*1	6400
11025	8	mono	11025*0.8*(8/8)*1	8820
22050	16	stereo	22050*0.8*(16/8)*2	70560
44100	16	stereo	44100*0.8*(16/8)*2	141120

2. Kuantisasi

Proses ini adalah proses pengkonversian nilai analog ke dalam suatu nilai diskrit. Selama proses kuantisasi, ADC menkonversi setiap nilai analog ke dalam bentuk diskrit. Diambil suatu input sinyal analog yang akan di kuantisasi, terlihat pada gambar III.4.



Gambar III.4. Input Sinyal Analog yang di Kuantisasi

Dari hasil kuantisasi padsa gambar III.4 maka dapat diambil data suara hasil sinyal diskrit dengan diwakili sebagai nilai *f(x)* sebagai berikut:

2	*3*	*4*	*4*
f(0)	*f*(1)	*f*(2)	*f*(3)

3. *Coding*

Coding, pada proses ini, tiap nilai diskrit yang telah didapat, direpresentasikan dengan angka *binary* n-bit. Cara konversinya adalah dengan membagi bilangan desimal dengan bilangan biner dengan memperhatikan hasil sisa pembagian.

Tabel III.2. Perhitungan Konversi Sinyal Diskrit ke Biner

Nilai diskrit dari hasil kuantisasi	Konversi ke biner	Kode biner (8-bit)
2	2:2 = 1 sisa 0 1:2 = 0 sisa 1	00000010
3	3:2 = 1 sisa 1 1:2 = 0 sisa 1	00000011
4	4:2 = 2 sisa 0 2:2 = 1 sisa 0 1:2 = 0 sisa 1	00000100
4	4:2 = 2 sisa 0 2:2 = 1 sisa 0 1:2 = 0 sisa 1	00000100

Kode biner hasil *coding* tidak akan dimasukan dalam perhitungan proses selanjutnya, karena kode biner ini hanya dibaca oleh *processor* komputer untuk pembacaan suatu sinyal digital.

III.1.2.2. Proses *Fast Fourier Transform* (FFT)

Fast Fourier Transform adalah suatu algoritma komputasi optimal yang mengimplemetasikan *Discreet Fourier Transform* (DFT) dengan teknik perhitungan yang cepat serta memanfaatkan sifat periodikal dari transformasi *fourier*. FFT merupakan operasi matematika yang bertujuan untuk dekomposisi dari suatu sinyal domain waktu ke sinyal domain frekuensi.

DFT dilakukan dengan mengimplementasikan sebuah transformasi, dengan panjang vektor *N* berdasarkan rumus :

$$F(u) = \frac{1}{N}\sum_{x=0}^{x=N-1} f(x) exp\ [-2j\pi ux/N] \qquad (3.2)$$

$$F(u) = \frac{1}{N}\sum_{x=0}^{x=N-1} f(x)\left(\cos\left(\frac{2\pi ux}{N}\right) - j\ sin\ \left(\frac{2\pi ux}{N}\right)\right) \qquad (3.3)$$

Misal diambil data suara hasil kuantitasi sinyal diskrit dengan nilai *f(x)* sebagai berikut:

2	*3*	*4*	*4*
f(0)	*f*(1)	*f*(2)	*f*(3)

nilai diskrit *f(x)* sebanyak 4 data, sehingga dapat ditentukan nilai *N* = 4 (banyak data), perhitungannya adalah:

Tabel III.3. Perhitungan DFT Sinyal Suara

N	***f(x)***	**Perhitungan DFT ($\sum_{x=0}^{N-1}$)**	**Hasil dari DFT / *F(u)***
1	*f(0)*	*1/4[2(cos(2*pi*0*0/4) – j sin(2*pi*0*0/4) + 3(cos(2*pi*0*1/4) – j sin(2*pi*0*1/4) + 4(cos(2*pi*0*2/4) – j sin(2*pi*0*2/4) + 4(cos(2*pi*0*3/4) – j sin(2*pi*0*3/4)]*	3.25
		1/4 [2(1-0) + 3(1-0) + 4(1-0) + 4(1-0)]	
2	*f(1)*	1/4[2(1-0) + 3(0- *j*) + 4(-1) + 4 (*j*)]	-0.5 + 0.25*j*
3	*f(2)*	1/4 [2(1-0) + 3(1-2) + 4(1-0) + 4(1-2)]	-0.25
4	*f(3)*	1/4[2(1-0) + 3(*j*) + 4(-1) + 4 (0-*j*)]	-0.5 - 0.25*j*

Perhitungan FFT mengimplementasikan pencerminan transformasi ganda hasil DFT dengan hanya menghitung nilai setengahnya data sinyal sehingga perhitungan akan lebih cepat, lalu nilai setengahnya lagi dihitung dengan cara *conjugate* nilai yang telah dihitung dengan DFT. Untuk membagi data sinyal adalah dengan fungsi (3.4) :

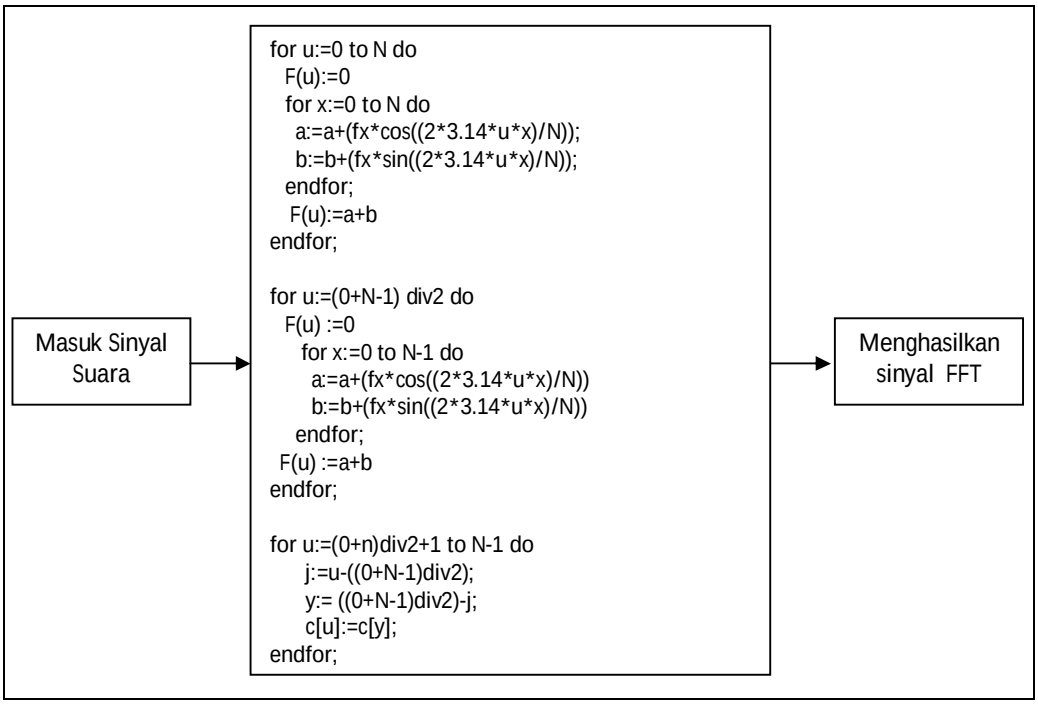
$$b = (N + 1) div 2 \qquad (3.4)$$

Berdasarkan fungsi (3.4) maka didapatkan $b = (4 + 1) div 2 = 3$, jadi perhitungan DFT hanya sampai data ke-3 / *f(2)* dan untuk data selanjunya hanya memakai fungsi *conjugate* dari nilai hasil DFT.

Tabel III.4. Perhitungan FFT dari hasil DFT

***f(x)* hasil**	**Hasil Perhitungan FFT / *F*(*u*)**	
f(0)	3.25	
f(1)	-0.5 + 0.25*j*	
f(2)	*-0.25*	← *nilai tengah (b)*
f(3)	-0.5 - 0.25*j*	← *hasil conjugate f(1)*
f(4)	3.25	← *hasil conjugate f(0)*

Alur masukan suara dimulai dari sinyal masuk, diproseses dengan rumus DFT, lalu setengah data hasil DFT di konjugasi agar menghasilkan nilai FFT.



Keterangan : Untuk hasil dari panjang sinyal FFT lebih detailnya dapat dilihat halaman lampiran.

Algoritma Procedure FFT

```
Procedure FFT(u: N:integer, a,b,fx:double, o: c:double)
{ Menghitung nilai Fast Fourier Transform
  IS : Banyaknya elemen data n dan sinyal suara
  FS : c[u] berupa elemen hasil FFT
}

Deklarasi
u, N      : integer
x, j , y  : double

Algoritma
for u←0 to N-1 do
   F(u) ←0
      for x←0 to N-1 do
      a←a+(fx*cos((2*3.14*u*x)/N))
        b←b+(fx*sin((2*3.14*u*x)/N))
     endfor x
  F(u)←a+b
endfor u

for u←(0+N-1)div2 do
   F[u]←0
      for x←0 to N-1 do
         a←a+(fx*cos((2*3.14*u*x)/N)
         b←b+(fx*sin((2*3.14*u*x)/N)
      endfor x
   F[u]← a+b
endfor u

for u←(0+N)div2+1 to N-1 do
   j←u-((0+N-1)div2)
   y←((0+N-1)div2)-j
   c[u]←c[y]
endfor u
```

III.1.2.3. Analisis *Speech Feature Extraction*

Untuk mencapai suatu *feature extraction*, akan digunakan metode analisis akustik untuk mengurangi suara menjadi beberapa set parameter dan teknik statistik untuk mengambil suara dan mengklasifikasikannya. Berikut adalah cara

mengukur nilai-nilai parameter fitur akustik dari berbagai suara masing-masing yang diambil dari *feature extraction* dan dijadikan sebagai sample pengenalan kata.

1. *Loudness* (Kekerasan Suara)

   Loudness diambil dari suara untuk mengukur tingkat kekerasan suara tiap-tiap data sampel suara. Sinyal didekati dengan perhitungan *Root Mean Square* (RMS). Dalam matematika, RMS dikenal sebagai rata-rata kuadrat, adalah ukuran statistik besarnya suatu kuantitas yang bervariasi. RMS ini berguna ketika terdapat variasi yang positif dan juga negatif, misalnya sinusoid. RMS digunakan dalam berbagai bidang, dan paling sering digunakan pada bidang sinyal. RMS dalam fitur ini menghitung RMS dalam domain frekuensi/FFT dengan perhitungan.

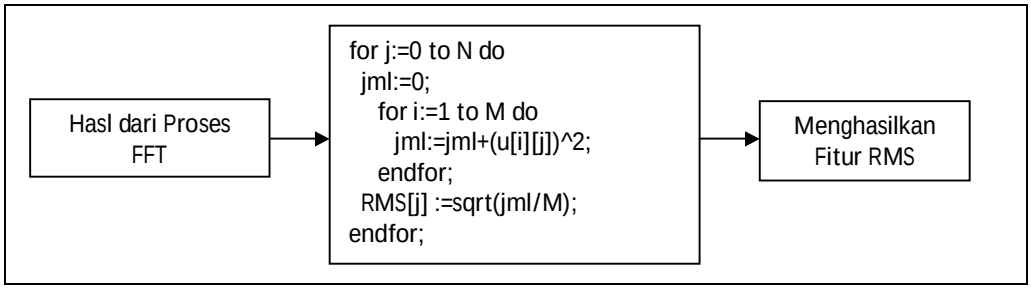
$$RMS_j = \sqrt{\frac{\sum_{i=1}^{M} |u_{ij}|^2}{M}} \quad , \; 1 \leq j \leq N \qquad (3.5)$$

Dari rumus (3.5) dapat diambil perhitungan seperti ini: (sum(u.*conj(u))/size(u))^1/2. Tabel III.6 berikut adalah perhitungan RMS untuk diambil fitur suara berdasarkan *loudness* :

Tabel III.5. Tabel Perhitungan Root Mean Square

Hasil FFT	**Perhitungan RMS**	**Hasil RMS**
3.25	(3.25.*conj(3.25)) / 5)	2.1125
-0.5 + 0.25*j*	(-0.5 + 0.25*j*).*conj(-0.5 + 0.25*j*)) / 5)	0.0625
-0.25	(-0.25.*conj(-0.25)) / 5)	0.0125
-0.5 - 0.25*j*	(-0.5 - 0.25*j*).*conj(-0.5 - 0.25*j*)) / 5)	0.0625
3.25	(3.25.*conj(3.25)) / 5)	2.1125
$\sum_{i=1}^{M}$	**(2.1125+0.0625+0.0125+0.0625+2.1125)^1/2**	**2.0887**

Alur masukan suara setelah dilakukan proses FFT lalu diambil nilai fitur RMS untuk ciri fitur dari kata tersebut.



Keterangan : Untuk hasil dari panjang fitur RMS lebih detailnya dapat dilihat halaman lampiran.

Algoritma Procedure Root Mean Square

```
Procedure rms(i: N:integer, u:double, o: RMS: double)
{
  Menghitung nilai ektraksi root mean square
  IS: Hasil FFT dan banyak elemen N
  FS: Rata-rata kuadrat sinyal FFT
}

Deklarasi
i     : integer
j,u,M : double

Algoritma
for j←0 to N do
  jml←0
     for i←1 to M do
        jml←jml+(u[i][j])^2
     endfor i
  RMS[j]←sqrt(jml/M)
endfor j
```

2. *Average Power Spectrum* (AVG)

Average Power Spectrum adalah proses untuk mengukur rata-rata daya dari sebuah sinyal periodik deterministik. Jenis sinyal kontinu dalam domain waktu, tetapi menghasilkan *discrete power spectrum*. Sebuah sinyal terdiri dari sinusoid contohnya sinyal listrik yang memiliki energi yang tak terbatas, tetapi rata-rata dayanya terbatas. Pertama-tama mengukur rata-rata daya menggunakan *periodogram spectrum object* dengan metode *window hamming*, dan menghitung rata-rata daya spektrum. Berikut rumus untuk metode *window hamming* pada sample ke-*N*.

$$W(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right),\ 0 \le n \le N-1 \qquad (3.6)$$

Tabel III.7 berikut adalah perhitungan *Hamming Window* untuk sample ke-n /*f*(0) dengan jumlah *N* = 4:

Tabel III.6. Perhitungan Windowing (Hamming)

Sampel ke-n / *f*(0) $0 \le n \le N-1$	**Perhitungan *window hamming***	**Hasil *window hamming***
2	0.54-0.46 cos((2*3.14*2) / 4-1)	0.619

Setelah proses *windowing* menggunakan *window hamming*, konversi nilai tersebut kedalam suatu nilai logaritmik.

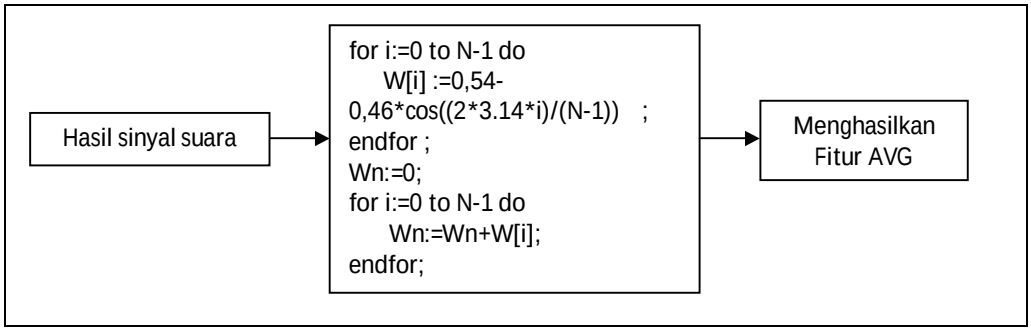
$$avgpower = 10 * log10\left(\frac{W(n)}{2}\right) \qquad (3.7)$$

Tabel III.8 berikut adalah adalah perhitungan untuk menghasilkan fitur *average power spectrum* :

Tabel III.7. Perhitungan AvgPower

$W(n)$ **(hasil window hamming)**	**Perhitungan AVG**	**Hasil AVG**
0.619	10*log10(0.619 / 2)	-5.0934

Alur masukan sinyal suara lalu diambil nilai fitur AVG untuk ciri fitur dari kata tersebut.



Keterangan : Untuk hasil dari panjang sinyal fitur AVG lebih detailnya dapat dilihat halaman lampiran.

Algoritma Procedure Average Power Spectrum

```
Procedure avgp(i: N:integer, o: avgp:double)
{
  Menghitung nilai average power spectrum
  Masukan: Banyaknya jumlah N dan sinyal f(x)
  Keluaran: Nilai logaritmik dari hasil window hamming
}

Deklarasi
i  : integer
Wn : double

Algoritma
for i←0 to N-1 do
      W[i]←0,54-0,46*cos((2*3.14*i)/(N-1))
      {menghitung hamming window}
endfor  i

Wn←0
```

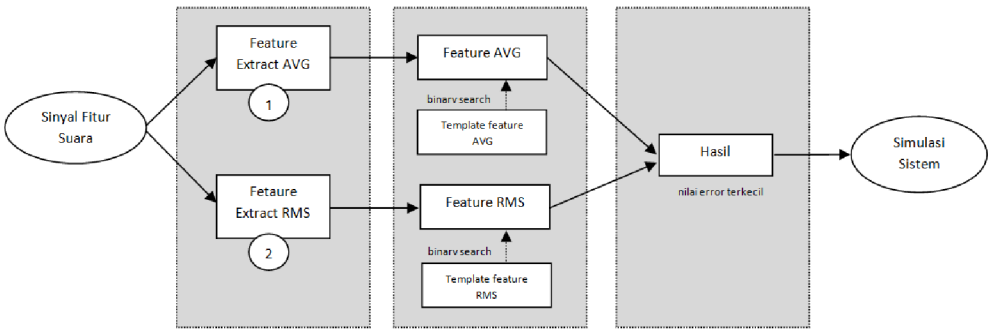
```
for i←0 to N-1 do
       Wn←Wn+W[i]
endfor i

avgp←10*log(10*(Wn/2)
```

III.1.3.Analisis Pengenalan Kata

Dalam sistem pengenalan kata menggunakan metode *divide and conquer*, ada beberapa tahap, yaitu membagi lebih dari satu proses ekstraksi dan pencocokan kata, dan saat pencocokan dilakukan algoritma *binary search divide and conquer*. Berikut adalah metode *divide and conquer* secara umum :

1) *Divide* : membagi masalah (proses ekstraksi) menjadi beberapa bagian proses yang lebih kecil.
2) *Conquer* : memecahkan atau menyelesaikan (pencarian biner) masing-masing sub proses/bagian menggunakan fungsi *binary search divide and conquer*.
3) *Combine* : menggabungkan solusi masing-masing proses sehingga memperoleh hasil dari solusi tersebut.



Gambar III.5. Proses Metode Divide and Conquer Pada Sistem Pengenalan Kata

Gambar III.5 menggambarkan proses dari metode *divide and conquer* yang bisa digunakan pada sistem pengenalan kata. Pertama adalah membagi proses ekstraksi fitur suara dalam beberapa proses, diawali dengan ektraksi fitur RMS dan seterusnya serta dilakukan pencocokan suara disetiap bagiannya, lalu dari setiap bagian diambil solusi atau hasil dari pencocokan masing-masing proses.

Pencocokan data suara baru dan hasil ekstraksi fitur pada template menggunakan teknik *binary search* implementasi metode *divide and conquer*.

III.1.3.1. Pencocokan Fitur *Loundness* Suara (RMS)

Misal didapatkan beberapa data hasil dari *feature extraction loudnesss (*RMS*)*:

3.0871	*2.0065*	*2.0433*	*3.0802*	*2.1167*	*2.0224*	*2.2765*	*2.0812*	*2.0058*

Sebelum dilakukan pencarian, data diurutkan dahulu agar pencarian lebih cepat. Hasil pengurutan data adalah sebagai berikut :

2.0058	*2.0065*	*2.0224*	*2.0433*	*2.0812*	*2.1167*	*2.2765*	*3.0802*	*3.0871*

Mula-mula diambil posisi awal $p = 1$ dan posisi akhir $q = N$, lalu cari posisi data tengah dengan rumus :

$$dt = (p + q)/2 \qquad (3.8)$$

Masuk satu data suara baru yang telah diekstraksi fitur *loudness*, dan dicari kecocokannya misal nilai yang akan dicocokan adalah : **2.0887**

dicari nilai tengah : $dt = (1 + 9)/2 = 5$

2.0058	*2.0065*	*2.0224*	*2.0433*	*2.0812*	*2.1167*	*2.2765*	*3.0802*	*3.0871*
↑				↑				↑
p				***dt***				***q***

karena 2.0887 > 2.0812 maka $p \rightarrow dt + 1 = 6$

dicari nilai tengah : $dt = (6 + 9)/2 = 7$

2.0058	*2.0065*	*2.0224*	*2.0433*	*2.0812*	*2.1167*	*2.2765*	*3.0802*	*3.0871*
					↑	↑		↑
					p	***dt***		***q***

karena 2.0887 < 2.2765 maka $q \rightarrow dt$; $dt - 1 = 6$

dicari nilai tengah : $dt = (6 + 6)/2 = 6$

2.0058	*2.0065*	*2.0224*	*2.0433*	*2.0812*	*2.1167*	*2.2765*	*3.0802*	*3.0871*
					↑ ↑ ↑			
					p dt q			

Karena nilai yang dicari adalah bilangan pecahan, maka satu data kiri *dt*, data *dt*, dan satu data kanan *dt* dilakukan fungsi pencarian nilai peramalan *error* terkecil *mean squared error(mse)* pada proses selanjutnya.

III.1.3.2. Pencocokan Fitur *Average Power Spectrum* Suara (AVG)

Didapatkan beberapa data hasil dari *feature extraction average power spectrum* (AVG):

-2.0752	*-1.0045*	*-1.0321*	*-1.0701*	*-1.0153*	*-1.0115*	*-1.1652*	*-1.0780*	*-1.0046*

Hasil pengurutan data adalah sebagai berikut :

-1.0045	*-1.0046*	*-1.0115*	*-1.0153*	*-1.0321*	*-1.0701*	*-1.0780*	*-1.1652*	*-2.0752*

Masuk satu data suara baru yang telah diekstraksi fitur *average power spectrum*, dan dicari kecocokannya misal nilai yang akan dicocokan adalah : **-1.0655**

dicari nilai tengah : $dt = (1 + 9)/2 = 5$

-1.0045	*-1.0046*	*-1.0115*	*-1.0153*	*-1.0321*	*-1.0701*	*-1.0780*	*-1.1652*	*-2.0752*
↑				↑				↑
p				***dt***				***q***

karena -1.0655 > -1.0321 maka $p \rightarrow dt + 1 = 6$

dicari nilai tengah : $dt = (6 + 9)/2 = 7$

-1.0045	*-1.0046*	*-1.0115*	*-1.0153*	*-1.0321*	*-1.0701*	*-1.0780*	*-1.1652*	*-2.0752*
					↑	↑		↑
					p	***dt***		***q***

karena -1.0655 < -1.0780 maka $q \rightarrow dt$; $dt - 1 = 6$

dicari nilai tengah : $dt = (6 + 6)/2 = 6$

-1.0045	*-1.0046*	*-1.0115*	*-1.0153*	*-1.0321*	*-1.0701*	*-1.0780*	*-1.1652*	*-2.0752*
					↑ ↑ ↑			
					p dt q			

Karena nilai yang dicari adalah bilangan pecahan, maka satu data kiri *dt*, data *dt*, dan satu data kanan *dt* dilakukan fungsi pencarian nilai peramalan *error* terkecil *mean squared error(mse)* pada proses selanjutya.

III.1.3.3. Menghitung *Mean Square Error* (MSE) Setiap Pencocokan Fitur

Perhitungan *mean squared error* untuk mencari nilai *error* terkecil adalah dengan rumus :

$$MSE - \frac{1}{N}\sum_{t=1}^{n} e(t)^2 \qquad (3.9)$$

Keterangan: $e(t) = a - b$

Tabel III.9 berikut adalah perhitungan dari peramalan *error* terkecil dari fitur *Loudness* (RMS) :

Tabel III.9. Perhitungan MSE Terkecil Fitur Loudness(RMS)

Nilai (*a*)	**Perhitungan error terkecil RMS (*b* = 2.0887)**	**Hasil *error(mse)* *RMS***	
2.2765	(2.2765 - 2.0887)^2	0.035268	
2.1167	(2.1167 - 2.0887)^2	0.000784	
2.0812	(2.0812 - 2.0887)^2	**0.0000562**	← *error terkecil*

Jadi data fitur RMS = 2.0887 dekat dengan nilai data 2.0812.

Tabel III.10 berikut adalah perhitungan dari peramalan *error* terkecil dari fitur *Average Power Spectrum* (AVG) :

Tabel III.10. Perhitungan MSE Terkecil Fitur Average Power Spectrum (AVG)

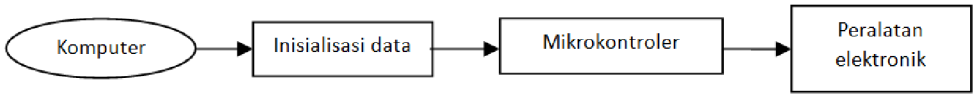
Nilai (*a*)	**Perhitungan error terkecil AVG (*b* = -1.0655)**	**Hasil *error(mse)* *AVG***	
-1.0321	((*-1.0321*) – (-1.0655))^2	0.056506	
-1.0701	((*-1.0701*) – (-1.0655))^2	**0.000021**	← *error terkecil*
-1.0780	((*-1.0780*) – (-1.0655))^2	0.000562	

Jadi data fitur AVG = -1.0655 dekat dengan nilai data -1.0701.

Begitupun pada proses pencarian di template fitur suara lain. Sehingga solusi atau hasil dari pengenalan diambil dengan membandingkan nilai *error* paling kecil pada perhitungan MSE setiap fitur. Dilihat dari nilai MSE paling kecil pada tabel perhitungan III.9 dan III.10 maka hasil pengenalan diambil dari fitur *average power spectrum* (AVG) dengan *error* terkecil sebesar 0.000021.

III.1.4. Analisis Komunikasi Serial Mikrokontroler

Analisis komunikasi serial mikrokontroler ini adalah kegiatan menganalisis bagaimana komputer agar dapat berkomunikasi dengan mikrokontroler menggunakan kabel serial dan mengaktifkan peralatan elektronik.



Gambar III.6. Alur Komunikasi Serial Komputer dan Mikrokontroler pada Peralatan Elektronik

1. Port Komunkasi Komputer

Pada komputer ada beberapa port yang bisa digunakan *port communication* . Komunikasi serial bisa melalui port USB dengan menggunakan kabel Serial-UART RS-232. Banyak port sesuai dengan jumlah port yang ada di setiap komputer, seperti port komunikasi (COM1). Port komunikasi pada komputer dapat di atur sesuai kebutuhan pengiriman data, kabel serial, dan mikrokontroler. Tabel III.8 berikut adalah spesifikasi standar *setting* port komunikasi yang ada di komputer.

Tabel III.8. Standar Setting Communication Port Komputer

Port Setting	***Value / Option***
Bit per second	1800, 2400, 4800, 7200, 9600, 14400, 19200, 38400, 57600, 115200, 128000
Data bits	4, 5, 6, 7, 8
Parity	*Even, Odd, None, Mark, Space*
Stop bits	1, 1.5, 2
Flow Control	*Xon / Xoff, Hardware, None*

2. Inisialisasi Data Perintah

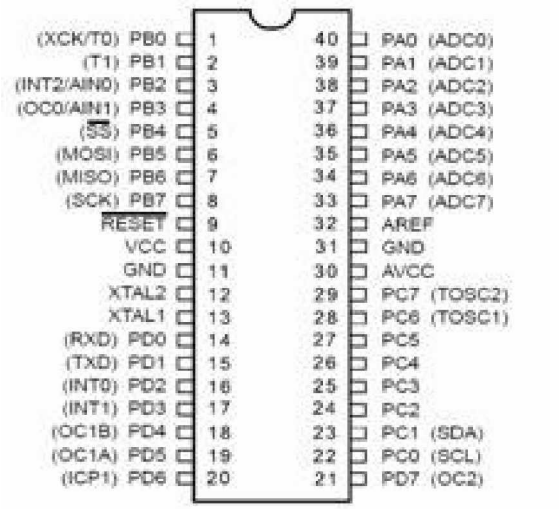
Pada bagian inisialisasi untuk input karakter sebagai perintah pada pemograman mikrokontroler menggunakan bahasa *asembly*. Mikrokontroler hanya mengenal inputan karakter ASCII berbentuk *hexadecimal*, berikut tabel karakter ASCII beserta nilai desimal dan heksadesimalnya.

Tabel III.9. Karakter ASCII

DEC	HEX	ASCII		DEC	HEX	ASCII
64	40	@		96	60	`
65	41	A		97	61	a
66	42	B		98	62	b
67	43	C		99	63	c
68	44	D		100	64	d
69	45	E		101	65	e
70	46	F		102	66	f
71	47	G		103	67	g
72	48	H		104	68	h
73	49	I		105	69	i
74	4A	J		106	6A	j
75	4B	K		107	6B	k
76	4C	L		108	6C	l
77	4D	M		109	6D	m

3. Arsitektur Pin Mikrokontroler

Agar dapat berkomunikasi antara komputer dengan mikrokontroler, perlu mengenal arsitektur dari mikrokontroler dan teknik komunikasi secara serial di komputer.



Gambar III.7. Arsitektur Pin Mikrokontroler ATmega8535

Tabel III.10 berikut adalah keterangan setiap fungsi Pin dari gambar arsitektur pin mikrokontroler Atmega8535 :

Tabel III.10. Fungsi Tiap-tiap Pin Mikrokontroler ATmega8536

Pin	Fungsi
VCC	Input sumber tegangan (+)
GND	*Ground* (-)
Port A (PA7 … PA0)	sebagai input analog dari ADC (*Analog to Digital Converter*). Port ini juga berfungsi sebagai port I/O dua arah, jika ADC tidak digunakan.
Port B (PB7 … PB0)	sebagai port I/O dua arah. Port PB5, PB6 dan PB7 juga berfungsi sebagai MOSI, MISO dan SCK yang dipergunakan pada proses *downloading*.
Port B (PC7 … PC0)	sebagai port I/O dua arah
Port B (PD7 … PD0)	sebagai port I/O dua arah. Port PD0 dan PD1 juga berfungsi sebagai RXD dan TXD, yang dipergunakan untuk komunikasi serial
RESET	*Input reset*
XTAL1	*Input* ke *amplifier inverting osilator* dan input ke *sirkuit clock internal*
XTAL2	*Output* dari *amplifier inverting osilator*
AVCC	Input tegangan untuk Port A dan ADC
AREF	Tegangan referensi untuk ADC

III.2. Analisis Kebutuhan Non-Fungsional

Analisis kebutuhan sistem dilakukan untuk mencari kebutuhan apa saja yang diperlukan untuk membuat sistem yang akan dibangun. Kebutuhan sistem yang diperlukan antara lain :

III.2.1. Kebutuhan Perangkat Keras

Kebutuhan minimum perangkat keras yang dibutuhkan untuk implementasi sistem yang akan dibuat. Adapun perangkat keras dibutuhkan pada aplikasi antara lain :

1. *Processor* Intel Pentium 4.
2. *Memory* 1GB.
3. *Harddisk* 60 GB
4. *Mikrophone mono*
5. *Speaker*
6. *Keyboard* dan *mouse*

Perangkat keras yang dibutuhkan pada simulasi sistem antara lain :

1. DT-AVR *Low Cost Micro System* dengan mikrokontroler ATmega8535
2. Kabel *Downloader K-125* Klinik Robot
3. Kabel serial UART RS-232
4. Rangkaian *driver motor*
5. *Prototype* peralatan elektronik (Motor DC, LED lensa, dan *slider* cd-rom)

III.2.2. Kebutuhan Perangkat Lunak

Pada bagian ini dijelaskan mengenai kebutuhan perangkat lunak dari aplikasi sistem pengenalan kata. Adapun perangkat lunak yang dibutuhkan pada aplikasi dan simulasi sistem antara lain :

1. Sistem Operasi

   Sistem Operasi yang digunakan adalah *Windows XP SP2*. Dipilih karena sistem operasi ini *user friendly* dengan aplikasi apapun.

2. Matlab

   Merupakan *interacvtive* program untuk *numerical computation* dan data *visualization* untuk analisis dan simulasi, dilengkapi dengan GUI matlab.

3. *Code Vision* AVR

   Adalah perangkat lunak pemrograman mikrokontroler keluarga AVR berbasis bahasa *Assembly* dan C.

4. *Driver serial* UART RS-232

   Adalah *software driver* untuk komunikasi antara komputer dengan DT-AVR *Low Cost Micro System* dan mikrokontroler.

III.2.3. Karakteristik Pengguna

Dalam menggunakan aplikasi ini dibutuhkan pengguna yang sudah berpengalaman dalam menggunakan komputer atau yang telah mengikuti *training* aplikasi ini dengan karakteristik sebagai berikut :

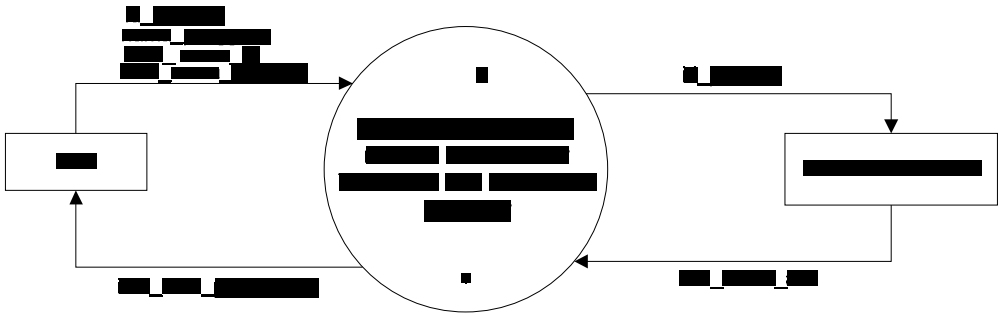
Pengguna	**Hak Akses**	**Tingkat Pendidikan**	**Tingkat Keterampilan**
umum	berinteraksi dengan aplikasi	minimal setara SMA	dapat menggunakan *software* Matlab

III.3. Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional pada sistem pengenalan kata ini meliputi diagram konteks dan *Data Flow Diagram* (DFD).

III.3.1. Diagram Konteks Sistem Perintah Suara Dengan Metode *Fast Fourier Transform dan Divide and Conquer*

Diagram konteks adalah diagram yang menunjukan keterhubungan antara perangkat lunak dengan konteks eksternal di luar program. Gambar III.8 berikut adalah diagram konteks dari sistem pengenalan kata menggunakan metode *divide and conquer*.

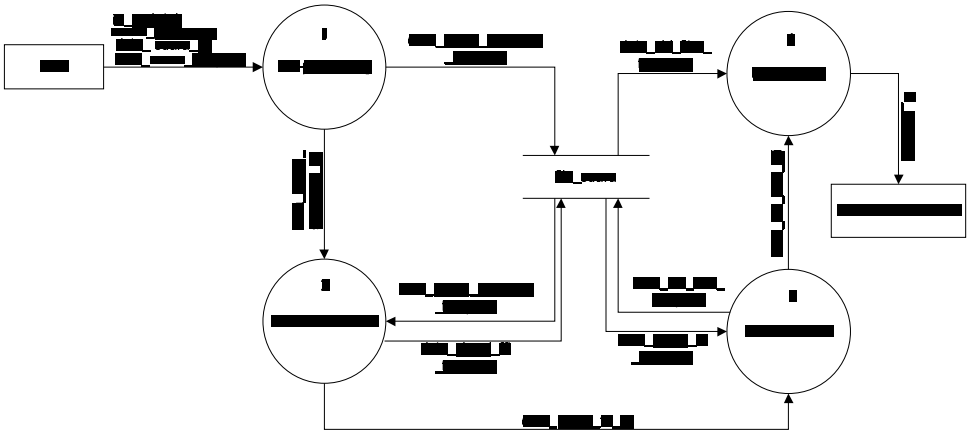


Gambar III.8. Diagram Konteks Sistem Perintah Suara Dengan Metode Fast Fourier Transform dan Divide and Conquer Pada Simulasi Rumah Pintar

III.3.2. *Data Flow Diagram* (DFD) Level 1 Sistem Perintah Suara Dengan Metode *Fast Fourier Transform* dan *Divide and Conquer*

Dari diagram konteks sebelumnya, maka selanjutnya proses diuraikan menjadi bagian-bagian yang lebih kecil. Dalam sistem pengenalan kata ini terdapat empat proses utama yaitu *pre-processing*, pemrosesan FFT, *feature extraction*, dan pengenalan kata. Empat proses tersebut berlaku saat perekaman

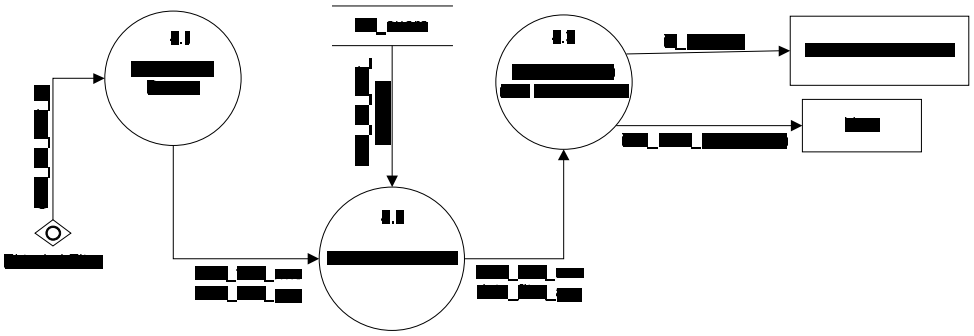
suara untuk template dan untuk pengujian pengenalan kata. Proses tersebut diuraikan dengan DFD level 1 pada gambar III.9.



Gambar III.9. DFD Level 1 Sistem Perintah Suara Dengan Metode Fast Fourier Transform dan Divide and Conquer Pada Simulasi Rumah Pintar

III.3.3. *Data Flow Diagram* (DFD) Level 2 Proses 4 (Pengenalan) Sistem Perintah Suara Dengan Metode *Fast Fourier Transform dan Divide and Conquer*

Proses pengenalan terbagi menjadi dua proses yaitu proses pembagian, pencocokan fitur, dan perhitungan nilai *error(mse)*. Masing-masing proses ini akan mengenali kata yang diucapkan dengan metode *divide and conquer*. Output dari proses ini adalah hasil pengenalan untuk pengujian dan perintah untuk mengaktifkan peralatan pada *prototype* simulasi rumah pintar.



Gambar III.10. DFD Level 2 Proses 4 Sistem Perintah Suara Dengan Metode Fast Fourier Transform dan Divide and Conquer Pada Simulasi Rumah Pintar

III.3.4. Spesifikasi Proses

Spesifikasi proses digunakan untuk mendeskripsikan dari setiap fungsi yang disajikan pada diagram alir data. Spesifikasi proses ditunjukan pada tabel III.11 :

Tabel III.11. Tabel Spesifikasi Proses

No	Proses	Keterangan
1	No.Proses	1
	Nama proses	*Pre-processing*
	Input	Suara berupa kata perintah yang direkam menggunakan mikrofon.
	Output	Sampling data sinyal diskrit domain waktu.
	Deskripsi	Proses ini digunakan untuk memproses sinyal suara yang telah direkam sebelum masuk ke proses selanjutnya
	Logika Proses	Sinyal suara diproses dengan cara *analog to digital conversion*, lalu dihasilkan data sampling sinyal diskrit berdomain waktu.
No	**Proses**	**Keterangan**
2	No.Proses	2
	Nama proses	Pemrosesan FFT
	Input	Sampling data sinyal diskrit domain waktu.
	Output	Sinyal domain frekuensi.
	Deskripsi	Proses ini dilakukan untuk mengubah data sinyal diskrit domain waktu menjadi data sinyal diskrit domain frekuensi menggunakan FFT.
	Logika Proses	Sampling sinyal diskrit domain waktu diproses dengan

		metode *Fast Fourier Transform*, metode ini merubah data sinyal domain waktu menjadi data sinyal domain frekuensi.
No	**Proses**	**Keterangan**
3	No.Proses	3
	Nama proses	Estraksi Fitur
	Input	Sinyal diskrit dan sinyal domain frekuensi.
	Output	Ciri fitur setiap suara.
	Deskripsi	Proses *feature extraction* dilakukan untuk mengambil ciri-ciri fitur dari setiap suara masukan.
	Logika Proses	Data sinyal diskrit dan sinyal domain frekuensi diolah dan di ambil dengan fitur *Loudness* (RMS) dan *Average Power Spectrum* (AVG), sehingga didapatkan dua ciri fitur setiap suara.
No	**Proses**	**Keterangan**
4	No.Proses	4
	Nama proses	Pengenalan
	Input	Ciri fitur setiap suara.
	Output	Hasil pengenalan dan perintah.
	Deskripsi	Proses pengenalan kata dilakukan dengan metode *divide and conquer*.
	Logika Proses	Data ciri fitur suara yang ada pada *template* dicocokan dengan data fitur suara baru dengan cara membagi proses masing-masing saat pencocokan dengan metode *Divide and Conquer*, lalu hasilnya membandingkan *error* yang paling kecil pada pencocokan setiap fiturnya, didapatkan hasil pengenalan dan perintah.
No	**Proses**	**Keterangan**
5	No.Proses	4.1
	Nama proses	Pembagian Proses
	Input	Ciri fitur setiap suara.
	Output	Fitur avg, dan rms
	Deskripsi	Hasil *feature extraction* dibagi menjadi dua bagian, yaitu *average power spectrum* (AVG) dan *root mean square* (RMS).
	Logika Proses	Ciri setiap fitur suara dibagi dua, ini berfungsi pada proses perbandingan nilai error terkecil untuk dijadikan hasil pengenalan.
No	**Proses**	**Keterangan**
6	No.Proses	4.2
	Nama proses	Pencocokan Fitur
	Input	Fitur AVG dan RMS.
	Output	Hasil pencocokan masing-masing fitur.
	Deskripsi	Setiap bagian fitur dilakukan pencarian mnilai yang cocok dengan *template feature* yang sudah ada menggunakan

		teknik *binary search divide and conquer*.
	Logika Proses	Ciri masing-masing fitur AVG dan RMS dicocokan antara ciri fitur suara baru dengan ciri fitur suara yang ada pada template dengan cara *binary search* teknik *divide and conquer* lalu dihitung nilai *error* terkecil setiap fitur dengan metode *mean squared error* (MSE).
No	**Proses**	**Keterangan**
7	No.Proses	4.3
	Nama proses	Perbandingan Nilai *Error* Terkecil
	Input	Hasil nilai *error* terkecil masing-masing fitur.
	Output	Hasil pengenalan dan perintah
	Deskripsi	Melakukan perbandingan nilai *error* terkecil dari fitur AVG dan RMS untuk dijadikan hasil pengenalan.
	Logika Proses	Nilai error terkecil fitur AVG dan RMS dibandingkan, dimana nilai *error* yang paling kecil adalah hasil dari pengenalan kata dan data perintah.

III.3.5.Kamus Data

Kamus data digunakan untuk menjelaskan semua data yang digunakan di dalam sistem. Tabel III.12 berikut adalah adalah kamus data untuk sistem yang dibangun.

Tabel III.12. Kamus Data

Nama Aliran Data :	nama_pengguna
Deskripsi	Nama pengguna yang memasukan suara untuk template
Tipe	*character*
Range	12
Nama Aliran Data :	id_perintah
Deskripsi	Kata dan id untuk perintah yang direkam, id perintah mewakili pemilihan dan info hasil kata yang diucapkan.
Tipe	*character*
Range	12
Nama Aliran Data :	data_sinyal_sampling_template
Deskripsi	Sampling data sinyal domain waktu untuk template.
Tipe	*double*
Range	5.0x10e-324 .. 1.7x10e308
Nama Aliran Data :	data_sinyal_sampling_uji
Deskripsi	Sampling data sinyal domain waktu suara baru.

Tipe	*double*
Range	5.0x10e-324 .. 1.7x10e308
Nama Aliran Data :	data_sinyal_fft_template
Deskripsi	Sinyal domain frekuensi untuk template yang didapat dari perhitungan FFT pada sampling
Tipe	*double*
Range	5.0x10e-324 .. 1.7x10e308
Nama Aliran Data :	data_sinyal_fft_uji
Deskripsi	Sinyal domain frekuensi suara baru yang didapat dari perhitungan FFT pada sampling
Tipe	*double*
Range	5.0x10e-324 .. 1.7x10e308
Nama Aliran Data :	data_ciri_fitur_template
Deskripsi	Ciri fitur tiap sinyal suara untuk template yang didapat dari proses *feature extraction*.
Tipe	*double*
Range	5.0x10e-324 .. 1.7x10e308
Nama Aliran Data :	data_ciri_fitur_uji
Deskripsi	Ciri fitur tiap sinyal suara baru yang didapat dari proses *feature extraction*.
Tipe	*double*
Range	5.0x10e-324 .. 1.7x10e308

III.4. Perancangan Sistem

Perancangan merupakan tahap lanjutan dari analisis sistem dimana pada perancangan sistem digambarkan rancangan sistem yang akan dibangun sebelum dilakukan pengkodean kedalam suatu bahasa pemograman. Dalam perancangan sistem tidak lepas dari hasil analisis, karena dari hasil analisis sistem baru dapat dibuat suatu perancangan sistem.

Desain umum yang akan diaplikasikan bertujuan untuk memberikan gambaran secara umum kepada penggunaan tentang sistem yang akan dibangun.

III.5. Perancangan File

Penyimpanan data yang digunakan berupa *file* yang bernama fftwav.mat, memiliki 5 field yaitu fft_name, fft_cmd, fft_wav, fft_db, dan fft_avg.

Tabel III.13. File Suara Sistem Perintah Suara

File fftwav.mat		
Nama	Tipe	*Range*
fft_name	char	12
fft_cmd	char	12
fft_wav	double	5.0x10e-324 .. 1.7x10e308
fft_db	double	5.0x10e-324 .. 1.7x10e308
fft_avg	double	5.0x10e-324 .. 1.7x10e308

III.6. Perancangan Arsitektur Menu

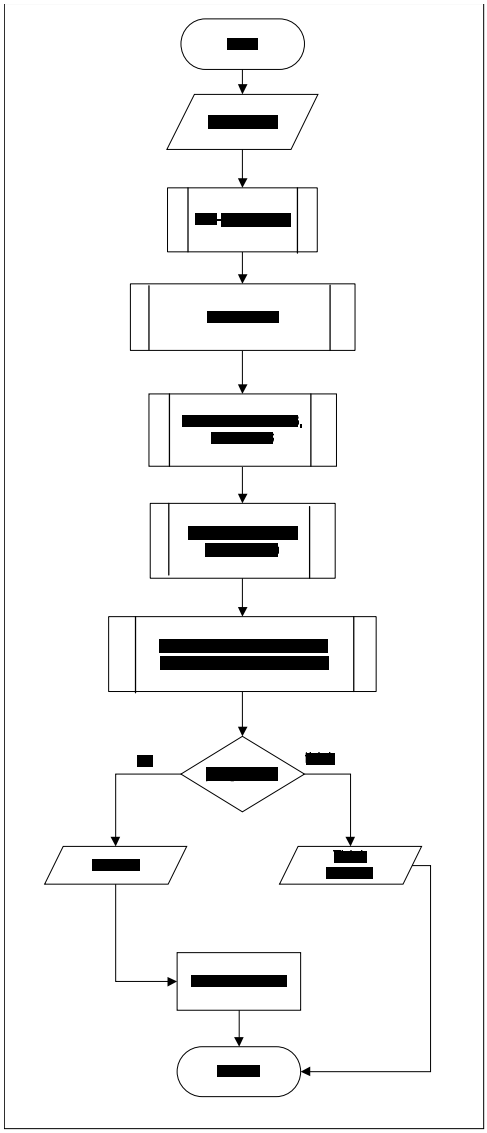
Perancangan arsitektur menu adalah perancangan dengan sebelumnya menggambarkan diagram alir sistem yang merupakan bagan yang menunjukan atas pekerjaan secara keseluruhan dari sistem yang akan dibuatkan menu pada aplikasi yang akna dibangun. Diagram ini menjelaskan urutan prosedur-prosedur yang ada dalam sistem. Diagram alir sistem menunjukan apa yang dikerjakan oleh sistem.

III.6.1.1. Diagram Alir Sistem Pengenalan Kata dan Simulasinya

Sinyal suara manusia berupa kata perintah yang diterima menggunakan *microphone* (sinyal analog) dicuplik/disampling sehingga menjadi sinyal digital. Proses pencuplikan sinyal ini dilakukan dengan menggunakan *soundcard* yang terdapat pada komputer. Pencuplikan dilakukan pada frekuensi 11025 Hz dengan resolusi bit 8 bit (1 byte) sehingga didapat data sebanyak 11025 byte per detik.

Kecepatan pencuplikan berdurasi 0,8 detik tiap kata berdasarkan penelitian sebelumnya.

Sinyal digital kemudian diproses dengan *pre-processing* sinyal yang menggunakan metode FFT sehingga didapatkan sinyal diskrit berdomain frekuensi. Sinyal hasil FFT tersebut di ekstraksi menggunakan fungsi AVG dan RMS untuk diambil fiturnya dan dibagi setiap prosesnya dengan metode *divide and conquer*. Setiap bagian proses masing--masing dilakukan pencocokan dengan tujuan untuk mempercepat proses pencocokan dengan fungsi *binary search* dan hasil dari masing-masing proses disatukan kembali dan diambil solusi untuk hasil pengenalan kata dan simulasi sistem.

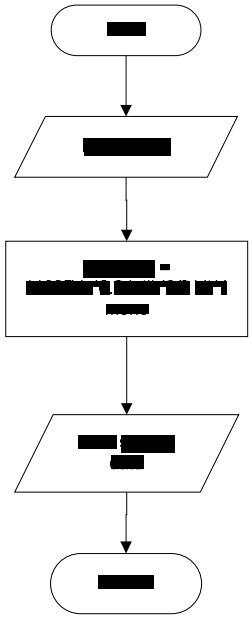


Gambar III.11. Flowchart Sistem Perintah Suara dan Simulasi

III.6.2. Tahap Sampling Suara

Suara direkam dan diberikan id perintah dengan format 1, dimana angka 1 menunjukan jenis kata perintah yang diucapkan dan berfungsi pada proses simulasi untuk mengirimkan perintah ke mikrokontroler. Berikut format id perintah setiap kata yang diucapkan; lampu on (1), lampu off (2), garasi buka (3), dan garasi tutup (4).

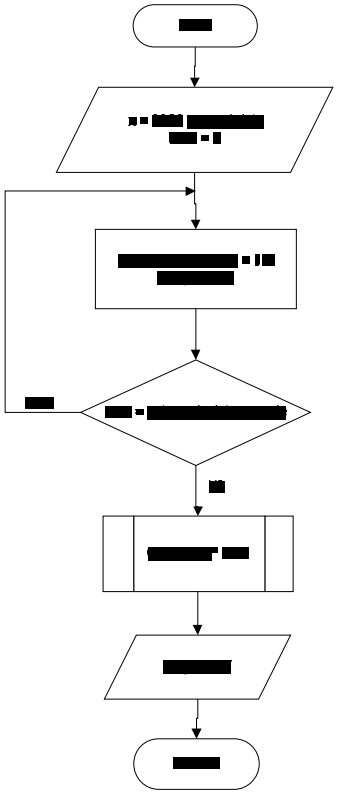
Data berupa sinyal suara diperoleh dengan cara merekam suara melalui mikrofon yang dihubungkan dengan komputer pribadi. Perekaman suara dilakukan dengan membuat fungsi *wave record* di dalam aplikasi dengan menggunakan bantuan program *accessories windows*, yaitu *sound recorder* dengan frekuensi sampling 11025Hz 8 bit mono. Frekuensi sampling sebesar 11025Hz sering digunakan untuk merekam suara manusia karena dinilai sudah cukup, dan kualitas suaranya kira-kira setara dengan lawan bicara ketika bicara di telepon. Lama suara yang akan direkam pada aplikasi adalah 0.8 detik, disampling dengan frekuensi sampling 11025Hz akan menghasilkan 8820 titik sampel data.



Gambar III.12. Flowchart Proses Sampling Suara

III.6.3.Tahap Pengolahan Sinyal FFT

Data sinyal yang dihasilkan dari proses sampling kemudian diolah menggunakan FFT untuk diambil data sinyal berdasarkan domain frekuensi. Berikut diagram alir dari proses FFT.



Gambar III.13. Flowchart Proses Pengolahan Sinyal Dengan FFT

Fungsi untuk FFT sendiri dapat langsung dipanggil di pemograman Matlab dengan contoh pemanggilan `Y = fft(x);` yang berarti Y adalah hasil dari FFT sinyal x.

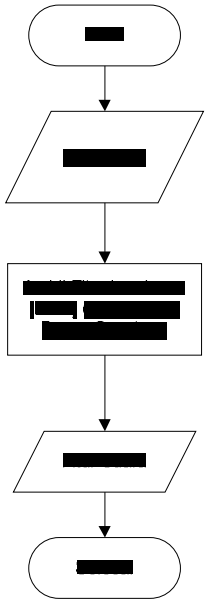
III.6.4.Tahap Ekstraksi Fitur

File suara yang sebelumnya direkam berekstensikan *wav* dan selanjutnya diproses kedalam suatu proses *sampling* dan proses FFT sehingga dapat dilakukan proses *feature extraction*, yaitu sebuah proses yang mengkonversi sinyal suara menjadi beberapa parameter yang dapat diambil untuk proses selanjutnya yaitu identifikasi pola suara.

Parameter yang memungkinkan dari suatu *file* suara adalah nilai amplitudo. Karena nilai amplitudo didapatkan berdasarkan domain waktu

sehingga untuk pengambilan sampel data akan didapatkan waktu yang mungkin berbeda-beda. Hal inilah yang menyebabkan suatu *feature extraction* tidak konsisten untuk dijadikan suatu acuan atau masukan pada proses selanjutnya.

Untuk itulah parameter dari suatu *file* suara tersebut harus diubah terlebih dahulu kedalam domain frekuensi dengan menggunakan *Fast Fourier Transform* karena pada manusia memiliki batas frekuensi tertentu sehingga data sampel bisa lebih konsisten untuk dijadikan suatu masukan pada proses selanjutnya. Proses inilah yang merupakan bagian dari pengolahan sinyal yang akan menghasilkan suatu *feature extraction* berupa nilai *magnitude* terhadap domain frekuensi.

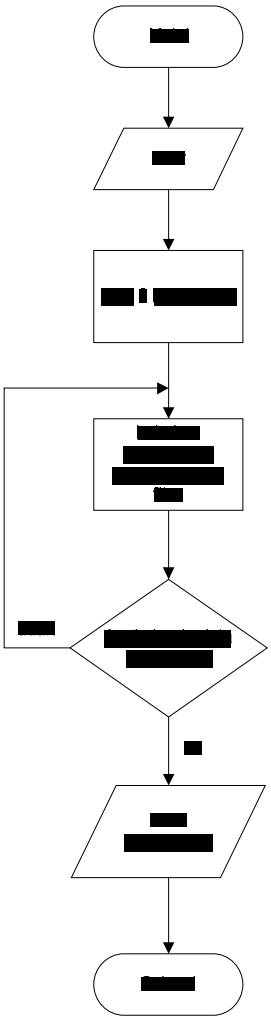


Gambar III.14. Flowchart Proses Feature Extraction

III.6.5. Tahap Pengenalan Kata

Pengenalan kata menggunakan metode *divide and conquer* pada program yang akan dirancang adalah dengan cara membagi proses ekstraksi sinyal suara menjadi 2 bagian, bagian pertama yaitu *root mean square* dan bagian kedua

average power spectrum. Pada setiap bagian dilakukan proses pencarian dengan menggunakan teknik *binary search divide and conquer* yang sebelumnya dilakukan pengurutan data terlebih dahulu. Setelah melakukan pencarian pada 3 bagian tersebut di ambil solusi dengan peramalan *error(mse)* terkecil untuk hasil dari pengenalan kata ini.



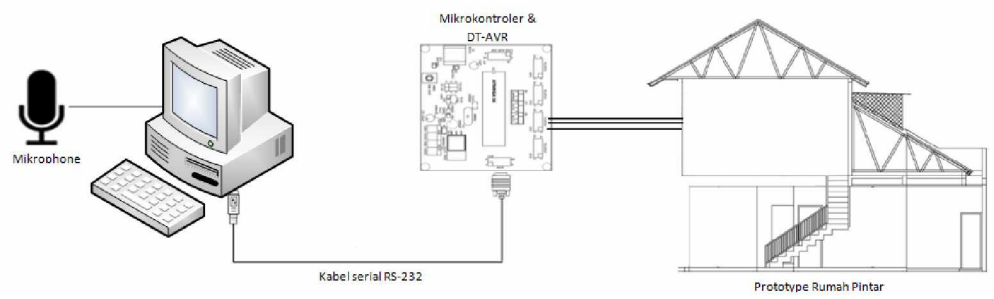
Gambar III.15. Flowchart Proses Pengenalan Kata

Untuk mendapatkan solusi atau hasil dengan nilai *error* terkecil, diperlukan perhitungan dengan fungsi *mean square error* (MSE). Sehingga

dengan nilai MSE terkecil, maka itulah solusi atau hasil dari pengenalan kata. MSE adalah rata-rata absolut dari kesalahan peramalan yang dikuadratkan. Fungsi MSE sendiri dapat langsung dipanggil oleh program Matlab.

III.6.6.Perancangan Perangkat Keras Untuk Simulasi

Proses simulasi sistem adalah proses pengujian aplikasi perintah suara dan implementasi pada alat yang akan dirancang di tahap ini. Sebelumnya sistem harus telah mengenali kata yang diucapkan oleh pengguna, setelah itu alat yang berkesesuaian dengan kata yang dikenali akan bereaksi pada *prototype* peralatan elektronik pada rumah pintar. Untuk dapat melakukan hal tersebut maka antara aplikasi dan mikrokontroler ATmega8535 yang terdapat di DT-AVR *Low Cost Micro System* harus dapat berkomunikasi yaitu dengan menggunakan kabel serial UART RS-232. Bahasa pemograman yang digunakan mikrokontroler dan DT-AVR *Low Cost Micro System* adalah bahasa *Assembly*.



Gambar III.16. Perancangan Perangkat Keras untuk Simulasi Pada Rumah Pintar

Pada aplikasi *port* yang digunakan adalah *port communication* (COM7), untuk setting *baud rate* akan menggunakan 9600 bps, jumlah bit data sebanyak 8 bit, jumlah *bit stop* sebanyak 1, dan input *buffer size*-nya 16000. Apabila pada proses pengujian dikenali ucapan (misal “Lampu On”) maka pada bagian *output*

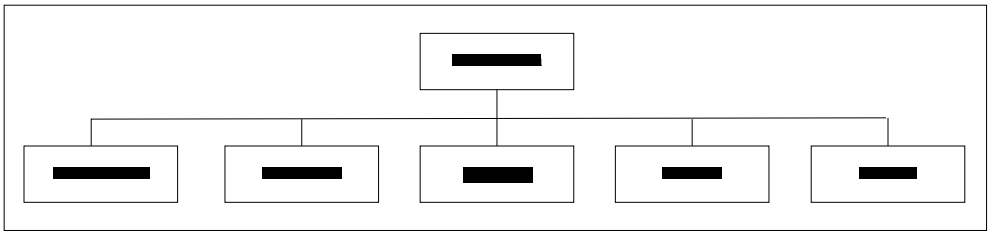
akan mengirimkan karakter ‘c’ dan pada mikrokontroler dan DT-AVR *Low Cost Micro System* akan menerima karakter ‘c’ dengan angka hexadesmal 0x63 sebagai perintah memberikan tegangan pada PORTA.1dan kemudian mengaktifkan alat yang terhubung dengan PORTA1. Tabel III.17 berikut adalah inisialisasi kata untuk data yang kan dikirim ke mikrokontroler dan mengaktifkan PORT pada mikrokontroler dan *prototype* peralatan elektronik :

Tabel III.14. Inisialisasi Kata Perintah

Kata Ucapan	Inisialisasi	Hexadesimal	Port
Lampu On	c	0x63	PORTA.6
Lampu Off	d	0x64	PORTA.6
Gerbang Buka	a	0x61	PORTA.2
Gerbang Tutup	b	0x62	PORTA.3

III.6.7.Perancangan Arsitektur Struktur Program

Perancangan arsitektur struktur program aplikasi perintah suara dengan *metode divide and conquer* pada simulasi rumah pintar yang akan dibangun.



Gambar III.17. Perancangan Arsitektur Struktur Menu Program

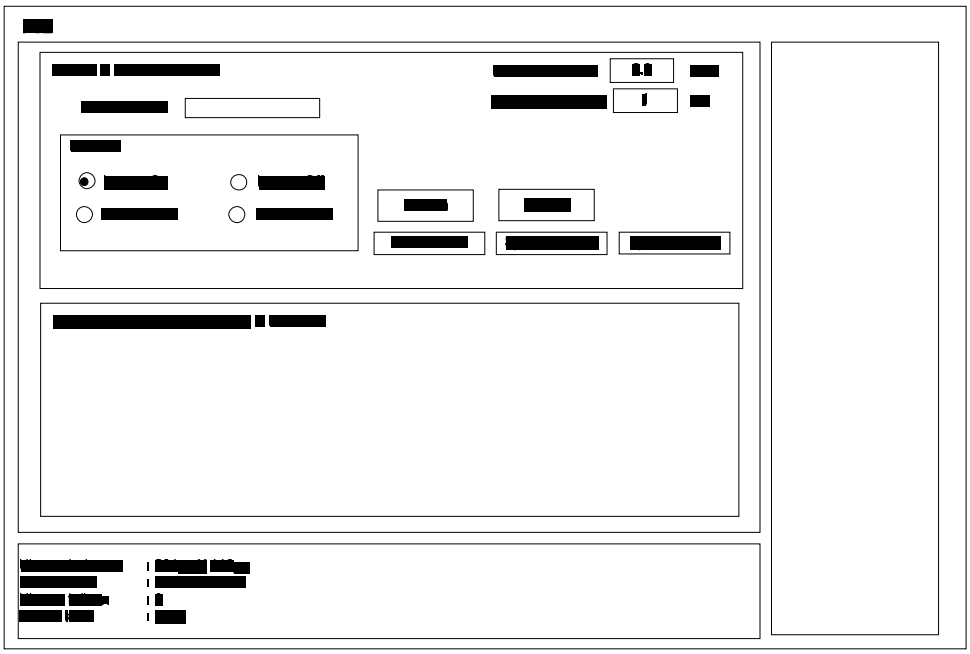
III.6.8.Perancangan Antarmuka

Perancangan antarmuka menu utama atau tampilan pertama kali pada aplikasi perintah suara dengan metode *Fast Fourier Transform* dan *Divide and Conquer* pada simulasi rumah pintar dengan lima pilihan menu, yaitu menu rekam suara, data suara, pengujian, simulasi dan bantuan.



Gambar III.18. Perancangan Antarmuka Menu Utama

Perancangan antarmuka menu rekam suara pada aplikasi perintah suara dengan metode *Fast Fourier Transform* dan *Divide and Conquer* pada simulasi rumah pintar.



Gambar III.19. Perancangan Antarmuka Rekam Suara

Perancangan antarmuka menu data suara pada aplikasi perintah suara dengan metode *Fast Fourier Transform* dan *Divide and Conquer* pada simulasi rumah pintar.



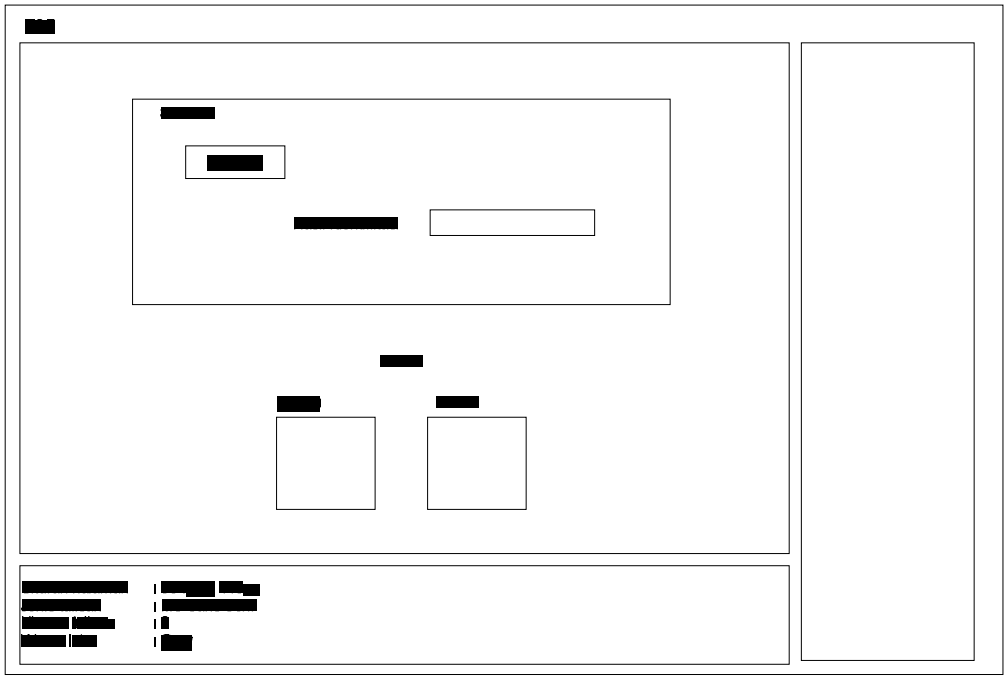
Gambar III.20. Perancangan Antarmuka Data Suara

Perancangan antarmuka menu pengujian pada aplikasi perintah suara dengan metode *Fast Fourier Transform* dan *Divide and Conquer* pada simulasi rumah pintar.



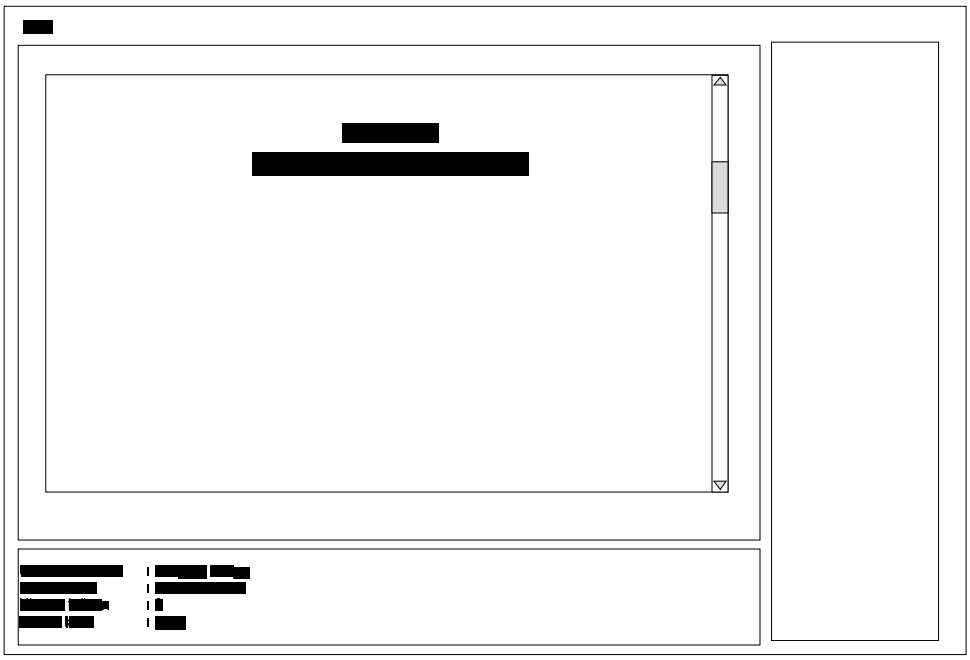
Gambar III.21. Perancangan Antarmuka Pengujian

Perancangan antarmuka menu simulasi pada aplikasi perintah suara dengan metode *Fast Fourier Transform* dan *Divide and Conquer* pada simulasi rumah pintar.



Gambar III.22. Perancangan Antarmuka Simulasi

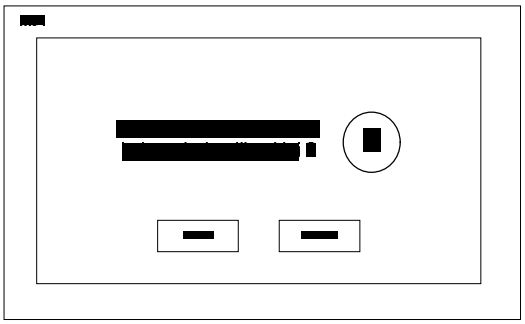
Perancangan antarmuka menu bantuan pada aplikasi perintah suara dengan metode *Fast Fourier Transform* dan *Divide and Conquer* pada simulasi rumah pintar.



Gambar III.23. Perancangan Antarmuka Bantuan

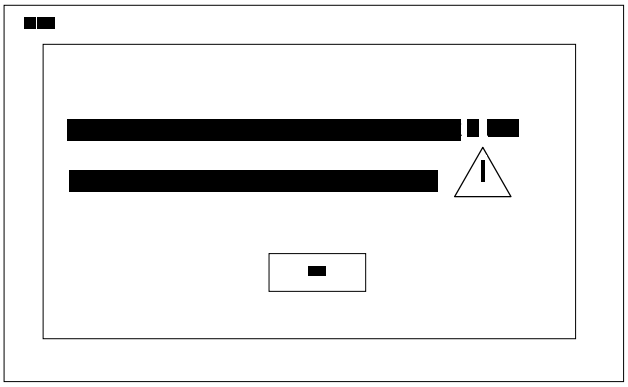
III.6.9. Perancangan Pesan

Perancangan pesan apabila pengguna keluar dari aplikasi.



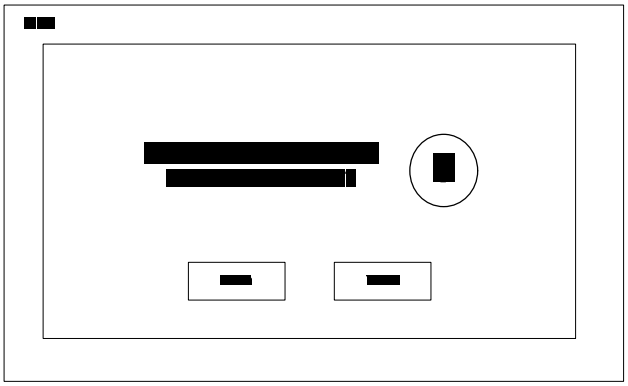
Gambar III.24. Konfirmasi Keluar Aplikasi

Perancangan pesan yang muncul apabila saat perekaman setiap kata belum mencapai 4 kali, lalu pengguna keluar dari menu rekam suara.



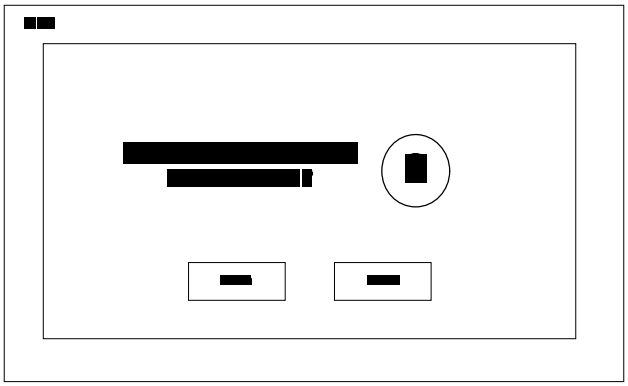
Gambar III.25. Pesan Rekam Kata Harus 4 Kali

Perancangan pesan yang muncul apabila pengguna mengklik tombol hapus semua pada menu data suara.



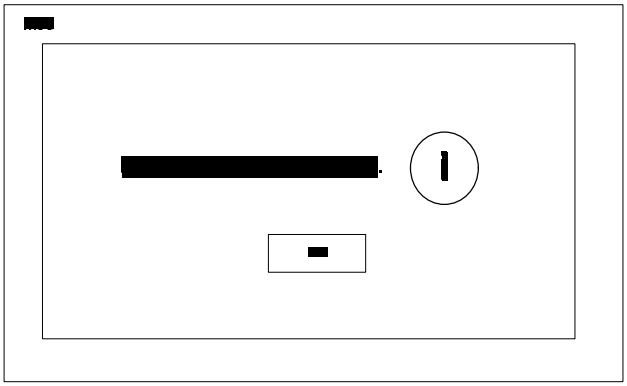
Gambar III.26. Konfirmasi Hapus Semua Data Suara

Perancangan pesan yang muncul apabila pengguna mengklik tombol hapus pada menu data suara.



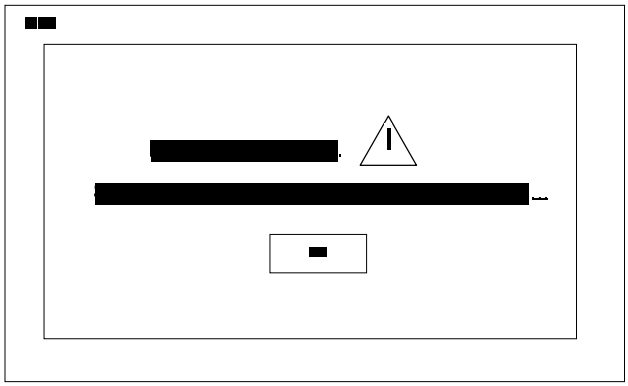
Gambar III.27. Konfirmasi Hapus Data Suara

Perancangan pesan yang muncul apabila pengguna mengklik tombol yakin pada pesan hapus data suara.



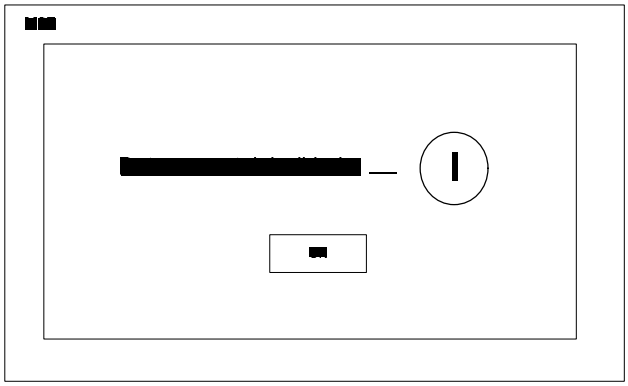
Gambar III.28. Pesan Data Suara Terhapus

Perancangan pesan yang muncul apabila pengguna mengklik tombol yang ada pada menu rekam suara saat data suara kosong.



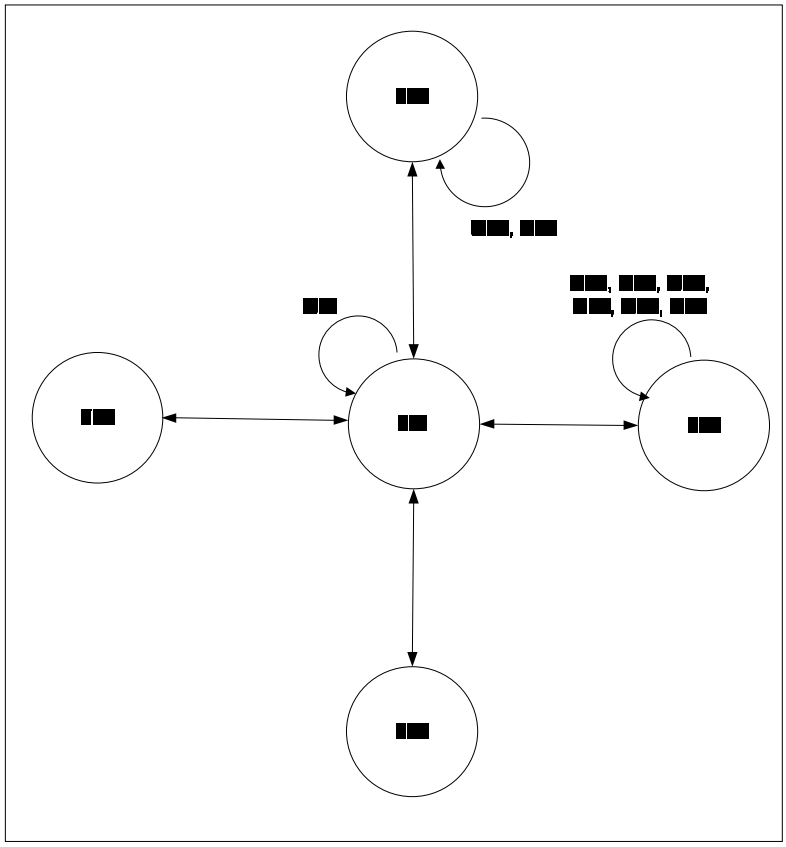
Gambar III.29. Pesan Data Suara Kosong

Perancangan pesan yang muncul apabila pengguna membuka file suara yang telah disimpan.



Gambar III.30. Pesan Data Suara Berhasil di Buka

III.6.10. Jaringan Semantik



Gambar III.31. Jaringan Semantik