DESAIN DAN IMPLEMENTASI *SOFTWARE RINGTONE COMPOSER* DAN *RINGTONE CONVERTER* PADA *HANDPHONE*

Kartika Gunadi, Yulia
Fakultas Teknologi Industri, Jurusan Teknik Informatika, Universitas Kristen Petra
e-mail : kgunadi@petra.ac.id , yulia@petra.ac.id

Tjandra Herry Prasetya
Alumni Fakultas Teknologi Industri, Jurusan Teknik Informatika
Universitas Kristen Petra

ABSTRAK: *Ringtone monophonic* mempunyai suatu pola atau bentuk penulisan yang berbeda antara jenis *handphone* yang satu dengan jenis *handphone* yang lainnya, sebagai contoh untuk Siemens c1(1/16) d1(1/8) sedangkan Nokia 16c1 8d1 sehingga untuk memasukkan *ringtone* jenis *handphone* Siemens ke jenis *handphone* Nokia diperlukan suatu konversi.

Deterministic Finite Otomata (DFA) dari teori bahasa dan otomata beserta metode logika dapat digunakan untuk melakukan konversi *ringtone*, dimana DFA digunakan untuk melakukan pemeriksaan terhadap *ringtone* yang diinputkan (dengan membuat *ringtone* sendiri (*compose*) atau membuka *ringtone* dari *text file*) dan metode logika digunakan untuk melakukan pengubahan pola atau penulisan *ringtone* dari jenis *handphone* yang satu ke jenis *handphone* lainnya dengan mengubah dan menyesuaikan pola penulisan ketukan, oktav dan nada *ringtone* sumber menjadi pola penulisan ketukan, oktav dan nada *ringtone* tujuan melalui proses iterasi sebanyak jumlah nada yang diinputkan.

Pada akhirnya program dalam penelitian ini dapat digunakan untuk mengonversi dan mengubah *ringtone* pada empat jenis *handphone* yaitu ericsson, nokia, samsung dan siemens.

Kata kunci: *Ringtone, Composer, Converter*

ABSTRACT: *Ringtone Monophonic have a different form in writing, depends on kind or type of its mobile phone. For example for Siemens c1(1/16) d1(1/8) but for Nokia 16c1 8d1 so, to input the ringtones from Siemens mobile phone to Nokia mobile phone needs a converter.*

Deterministic Finite Automata (DFA) from language and automata theory with logical method can be used to convert ringtone which DFA will be used to check the ringtone that had been entered (it can be compose by own or open it from a text file) and logical method is used for changing pattern or writing ringtone between different kinds of mobile phone by changing and fiting the writing pattern of the tap, octave and source ringtone to be writing pattern of the tap, octave and target ringtone through the iteration process as much as the ringtone had been entered.

Finally, this research software can be used to convert and compose the ringtone for four different types of mobile phone such as Ericsson, Nokia, Samsung and Siemens.

Keywords*: Ringtone, Composer, Converter.*

1. PENDAHULUAN

Format penulisan *ringtone* yang ada saat ini dapat dikategorikan menjadi dua bagian: *text base* dan *graphic base. Graphic base* lebih dikenal dengan notasi balok.

Format penulisan *ringtone* dengan menggunakan *text base* tidak mempunyai standard penulisan yang pasti, berbeda dengan format penulisan *ringtone* menggunakan *graphic base* yang mempunyai standard penulisan not balok yang meliputi garis paranada, simbol not balok beserta nilai ketukannya.

Karena tidak adanya standard penulisan yang pasti pada format *text base*, maka format penulisan *ringtone* menggunakan *text base* menimbulkan perbedaan antara *vendor* yang satu dengan yang lainnya.

Untuk mempermudah pembuatan *ringtone* dan konversi *ringtone* bagi orang yang

tidak mengerti format penulisan *ringtone*, maka diperlukan suatu *software* yang dapat memeriksa format penulisan *ringtone* pada suatu jenis *handphone* dan dapat melakukan konversi format penulisan *ringtone* dari jenis *handphone* yang satu ke jenis *handphone* yang lainnya dan konversi format penulisan *ringtone* ke pola penekanan tombol pada setiap jenis *handphone*.

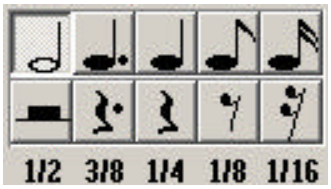
2. *RINGTONE*

Ringtone merupakan salah satu fasilitas yang terdapat pada *handphone*, yang berfungsi untuk memberitahukan pemakai *handphone* jika ada panggilan yang masuk. *Ringtone* dibagi menjadi dua macam yaitu *monophonic* dan *polyphonic*. *Monophonic* mempunyai arti bahwa suara alat musik yang dapat dibunyikan secara hanya ada satu jenis suara alat musik saja, sedangkan *polyphonic* lebih dikenal dengan 16 *polyphonic* mempunyai arti bahwa suara alat musik yang dapat dibunyikan secara bersamaan jenisnya ada 16 macam alat musik, antara lain alat musik biola, bass, terompet dan sebagainya sehingga suara yang dihasilkan menyerupai suara orkestra.

Pada model *handphone* tertentu antara lain Nokia 3210, Siemens C35 dan Siemens M35 mempunyai jumlah nada yang terbatas yaitu maksimum sebanyak 50 buah nada yang berarti suara atau *ringtone* yang dimasukkan ke dalam model *handphone* tersebut harus memiliki jumlah nada maksimum sebanyak 50 buah.

Setiap vendor mempunyai format penulisan *ringtone* yang berbeda:

1. Siemens :
   [Nada] [(Tanda Kromatis)] [(Oktav)] [(Ketukan Nada)], dimana :
   Nada : c, d, e, f, g, a, h, c', p
   Tanda Kromatis : # (kres) disimbolkan dengan 'is'
   Oktav : 1, 2, 3, 4
   Ketukan Nada : 3/1, 2/1, 1/1, 1/2, 1/4, 1/8, 1/16 dalam program ketukan 3/1 & 2/1 tidak digunakan karena pada dasarnya *ringtone* yang ada hanya menggunakan ketukan mulai dari 1/1 sampai dengan 1/16
2. Nokia :
   [(Ketukan Nada)] [(Spesial Ketukan Nada)] [(Tanda Kromatis)] [Nada] [(Oktav)], dimana :
   Ketukan Nada : 1,2,4,8,16,32
   Spesial Ketukan Nada : .
   Tanda Kromatis : # (kres)
   Nada : c,d,e,f,g,a,b,c',-
   Oktav : 1,2,3
3. Ericsson :
   [(Oktav)] [(Tanda Kromatis)] [Nada & (Ketukan Nada)], dimana :
   Oktav : Terdiri dari 2 oktav (oktav 2 disimbolkan tanpa menggunakan tanda "+", oktav 3 disimbolkan dengan menggunakan tanda "+")
   Tanda Kromatis : b (mol), # (kres) dalam program hanya menggunakan tanda kromatis # (kres)
   Nada & Ketukan Nada : c,d,e,f,g,a,b,c',p
   Ketukan nada disimbolkan menggunakan huruf besar dan huruf kecil, contoh : C mempunyai ketukan 1/8, c mempunyai ketukan 1/16
4. Samsung :
   Sama dengan format Nokia (kompatibel dengan Nokia), hanya saja di tampilkan dalam format not balok, dengan simbol not balok seperti pada gambar berikut :

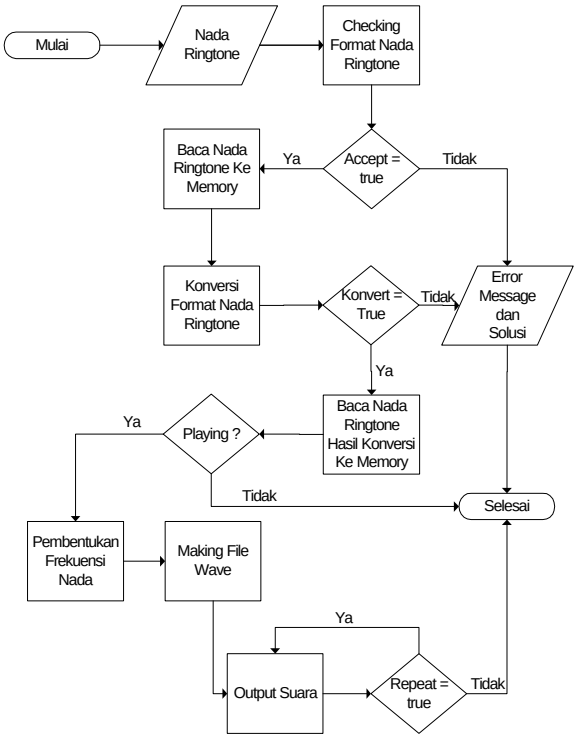


Gambar 1. Nada dan Ketukan Nada pada Samsung

Gambar 2. Tanda Kromatis pada Samsung

3. DESAIN DAN IMPLEMENTASI SISTEM

Berikut ini adalah *flowchart* umum program aplikasi untuk proses konversi dan memainkan *ringtone* :



Gambar 3. *Flowchart* Umum Program Aplikasi

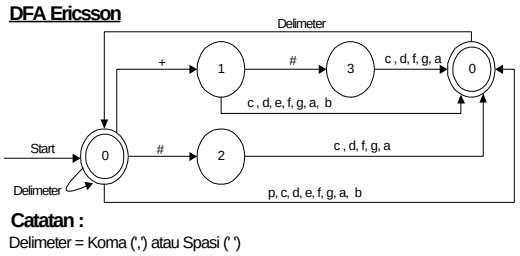
Sistem program terbagi menjadi tujuh bagian: penting dengan *input* dan *output* maing-masing:

1. *Editor* yang berfungsi sebagai *inputan*
   Editor merupakan lembar kerja untuk memasukkan nada. Terdapat dua macam *editor*, yaitu *editor* untuk *grammar* yang berbentuk *text base* (untuk semua jenis *handphone*) dan *editor* untuk *grammar* yang berbentuk notasi balok atau *graphic base* (khusus untuk jenis *handphone* Samsung). *Editor* untuk *grammar* yang berbentuk huruf dibagi menjadi dua macam, yaitu:
   - *Source Editor*
     Berfungsi untuk menampung masukan *grammar ringtone* dari salah satu jenis *handphone* baik dari proses membuka *text file, paste* dari *clipboard* atau penulisan *grammar ringtone* sendiri sebelum dikonversikan ke *grammar ringtone* jenis *handphone* yang lainnya. Penulisan *grammar ringtone* sendiri pada *source editor* dikenal dengan istilah "*compose*". Pada *source editor* dapat dilakukan proses *grammar editing* antara lain meng-*copy grammar* ke *clipboard*, menghapus, mengubah, menyisipkan dan menambah *grammar*.
   - *Target Editor*
     Berfungsi untuk menampung hasil konversi *grammar ringtone* dari *source editor* dan menyimpan *grammar* hasil konversi ke dalam *text file*.
2. Pembacaan dan pemeriksaan *grammar* (*scanner*) dari *editor*
   Pada bagian ini terjadi proses pembacaan dan pengartian serta pemeriksaan bentuk *grammar* berdasarkan jenis *ringtone* pada masing-masing *handphone* yang diambil dari *source editor*. Hasil proses ini berupa nada, ketukan, spesial ketukan, oktav dan *delimeter*, yang disimpan dalam *array* berdimensi satu dimana maksimalnya adalah seratus. Metode yang dipakai dalam pengartian *grammar* adalah metode logika yang disesuaikan dengan pola penulisan *ringtone* pada *composer* masing-masing jenis *handphone*. Sedangkan *scanner* berfungsinya untuk melakukan pemeriksaan terhadap *grammar* (*inputan*).

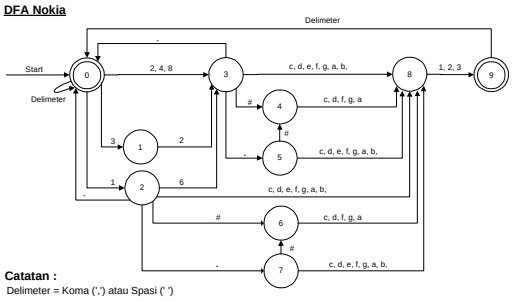
Syarat-syarat membuat DFA (*Deterministic Finite Automata*) adalah :

1. Tidak mempunyai *empty string*
2. Tiap *state* tidak boleh mempunyai *inputan* yang sama

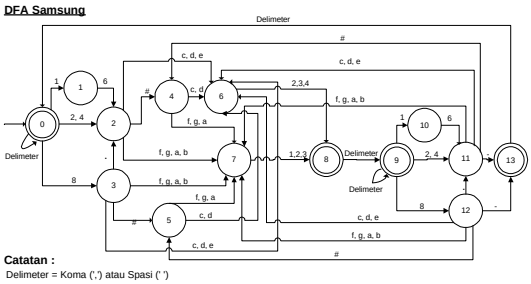
DFA (*Deterministic Finite Automata*) dari masing-masing jenis *handphone* dapat dilihat pada gambar 4, 5, 6 dan 7.



Gambar 4. DFA Ericsson

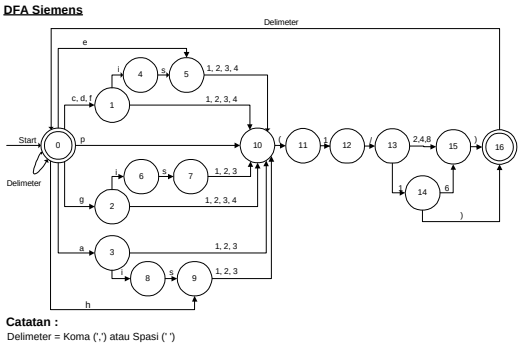


Gambar 5. DFA Nokia



Gambar 6. DFA Samsung

Pada dasarnya *composer ringtone* pada *handphone* Samsung menggunakan notasi balok (*graphic base*) dan bukan *text base* sehingga DFA untuk *handphone* Samsung dalam aplikasi ini dirancang hampir menyerupai DFA untuk *handphone* Nokia. Tujuan perancangan *grammar* Samsung adalah untuk memudahkan proses-proses lainnya.



Gambar 7. DFA Siemens

3. Menggambar ulang nada pada *editor* not balok
   Bagian ini akan menghasilkan gambar not balok beserta dengan tanda – tandanya pada saat dilakukan proses konversi, proses membuka *grammar* dari *text file*, proses menghapus atau menyisipkan not balok pada *editor* not balok dan proses *scrolling* pada *editor* not balok jika *handphone* sumber atau *handphone* tujuan yang dipilih adalah Samsung.
4. Pembuatan *wave file*
   Bagian ini untuk mengubah *grammar* yang ada pada *target editor* menjadi *wave file*. Pembuatan *wave file* dilakukan dalam proses *playing* (dimulai ketika *user* menekan tombol *play*). *Wave file* dibuat dengan menggunakan frekuensi 22050 Hz, 16 bits per sample dan *wave channel* maksimum adalah empat, yang berarti *wave file* yang akan dihasilkan nantinya memiliki kualitas suara sama dengan suara radio dan *mode* suara yang dipakai adalah mono. *Wave file* yang dibuat menggunakan standar Microsoft *format* yang memiliki tiga potong bagian informasi yaitu RIFF *chunk*, format *chunk* dan data *chunk*. RIFF *chunk* berfungsi untuk menyatakan bahwa suatu *file* adalah *wave file*, *format chunk* berfungsi untuk menyatakan parameter dari *wave file* seperti *sample rate*, bits per *sample* dan sebagainya, sedangkan data *chunk* untuk menyatakan data *wave file*.
5. *Playing*
   Bagian ini berfungsi untuk memainkan *ringtone* setelah yang telah diubah menjadi *wave file*. *Wave file* dimainkan dengan menggunakan komponen TMediaplayer pada Delphi dan TEvent pada Delphi untuk melakukan pengulangan ketika *wave file* dimainkan.
6. Konversi
   Bagian ini untuk mengubah *format ringtone* dari jenis *handphone* yang satu ke jenis *handphone* yang lainnya. Proses konversi meliputi empat tahap, yaitu :
   1. Pemeriksaan *grammar* sumber
   2. Pembacaan *grammar* sumber ke memory
   3. Konversi
   4. Pembacaan *grammar* hasil konversi ke memory

   Dalam melakukan proses konversi perlu diperhatikan satu hal penting yaitu *range* nada, karena *range* nada pada setiap jenis *handphone* tidak sama. Jika *range* nada pada *ringtone* sumber tidak memenuhi *range* nada pada *ringtone* tujuan, maka akan dilakukan penyesuain *range* nada pada *ringtone* sumber dengan menaikkan atau menurunkan

nilai oktav dan ketukan pada nada sebelum dilakukan proses konversi.

7. *Step menu*
   Bagian ini dibagi menjadi tiga macam *step menu*. *Step menu* yang pertama mempunyai fungsi untuk menentukan format asal *ringtone* berdasarkan jenis *handphone*, *step menu* yang kedua mempunyai fungsi untuk menentukan tujuan konversi format *ringtone* berdasarkan jenis *handphone*, sedangkan *step menu* yang ketiga mempunyai fungsi untuk menentukan tampilan akhir hasil konversi baik berupa *keypresses* atau sesuai format *ringtone* asli (*composer*). Hal ini ditujukan untuk memudahkan proses memasukkan *ringtone* ke dalam *handphone*.

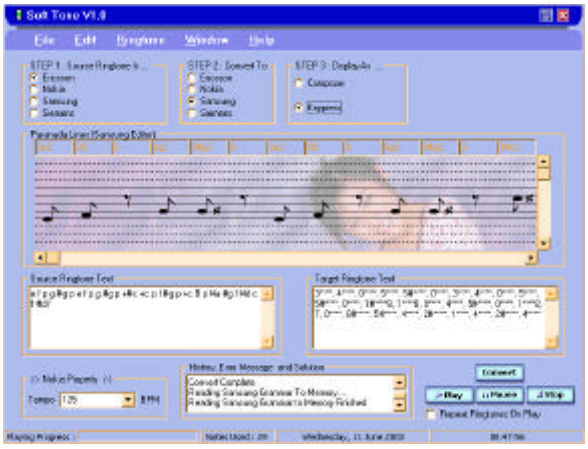
4. PENGUJIAN

4.1 Konversi

Pengujian Ericsson *to* Samsung *display as keypress*

- *Source ringtone is …* : Ericsson
- *Convert to* : Samsung
- *Display as …* : *Keypress*
- Tempo : *Default* (160)

Hasil pengujian dapat dilihat pada gambar 8



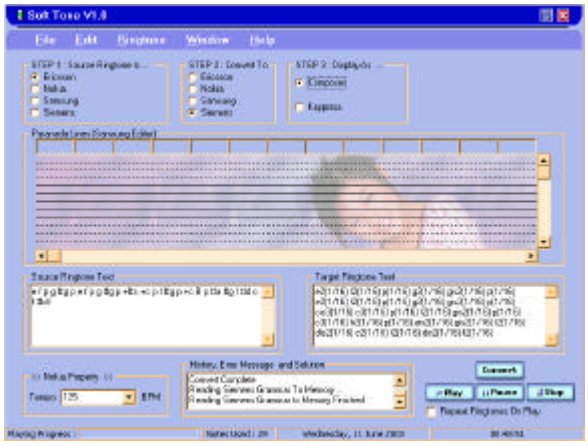
Gambar 8. Hasil Pengujian Ericsson *to* Samsung *Display As Keypress*

Pada hasil pengujian terlihat bahwa pada *source ringtone text editor* yang berisi *ringtone* Ericsson dengan *grammar* e f p g #g dan seterusnya jika dikonversikan mengalami perubahan (bandingkan antara *source ringtone text editor* dengan *target ringtone text editor*) karena konversi dilakukan pada *handphone* yang tidak sejenis yaitu dari Ericsson ke Samsung. Hasil akhir konversi ditampilkan dalam *keypresses* dan juga ditampilkan dalam bentuk not balok. Format *ringtone* yang awalnya e f p g #g dan seterusnya menjadi 3****, 4****, 0****, 5****, 5#**** dan seterusnya.

Pengujian Ericsson *to* Siemens *display as composer*

- *Source ringtone is …* : Ericsson
- *Convert to* : Siemens
- *Display as …* : *Composer*
- Tempo : *Default* (120)

Hasil pengujian dapat dilihat pada gambar 9



Gambar 9. Hasil Pengujian Ericsson *to* Siemens *Display As Composer*

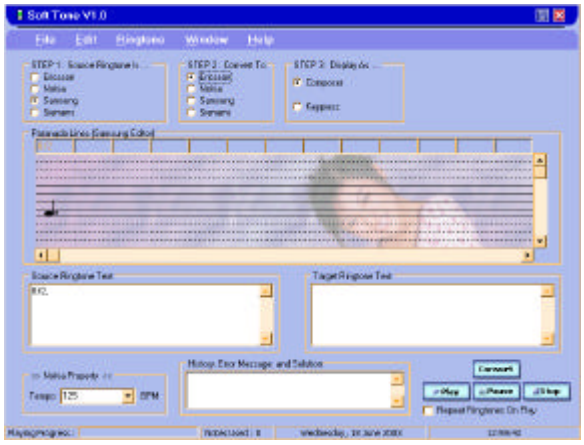
Pada hasil pengujian terlihat bahwa pada *source ringtone text editor* yang berisi *ringtone ericsson* dengan *grammar* e f p g #g dan seterusnya jika dikonversikan mengalami perubahan (bandingkan antara *source ringtone text editor* dengan *target ringtone text editor*) karena konversi dilakukan pada *handphone* yang tidak sejenis yaitu dari Ericsson ke Siemens. Format *ringtone* yang awalnya e f p g #g dan seterusnya menjadi e2(1/16) f2(1/16) p(1/16) g2(1/16) gis2(1/16) dan seterusnya.

4.2 Editor Not Balok

Pengujian dilakukan dengan memasukkan not balok pada *editor* not balok (untuk

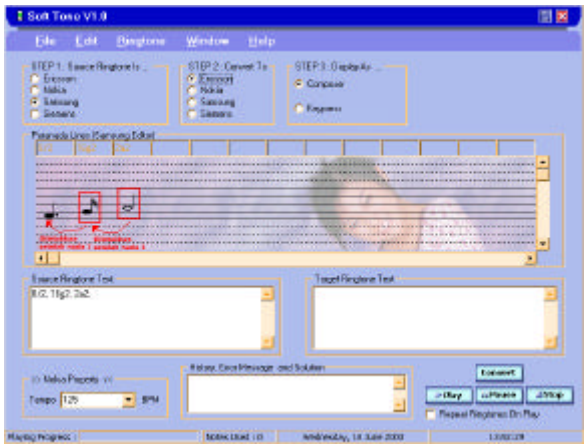
compose atau membuat nada *ringtone* sendiri dan *editing*) :

- Pilih tangkai nada, nada diam atau tanda kromatis pada form not balok
- Arahkan *mouse* pada *editor* not balok sesuai posisi nada yang dikehendaki.
- Letakkan nada pada *editor* not balok dengan menekan tombol kiri *mouse*. Program akan mengetahui apakah nada harus disisipkan atau tidak. Jika *editor* kosong (belum berisi nada sama sekali) letakkan nada pada posisi awal *editor* kemudian untuk nada berikutnya letakkan tepat dibelakangnya. Untuk menyisipkan nada, letakkan nada pada salah satu *editor* yang sudah berisi nada tepat diatasnya, maka secara otomatis nada yang dibawahnya akan bergeser ke kanan. Gambar 11 menunjukkan hasil peletakkan nada pertama pada posisi awal *editor* sedangkan gambar 12 menunjukkan hasil peletakkan nada kedua tepat dibelakang nada pertama dan menunjukkan hasil peletakkan nada ketiga tepat dibelakang nada kedua. Gambar 13 menunjukkan hasil penyisipan nada diantara nada kedua dan ketiga.

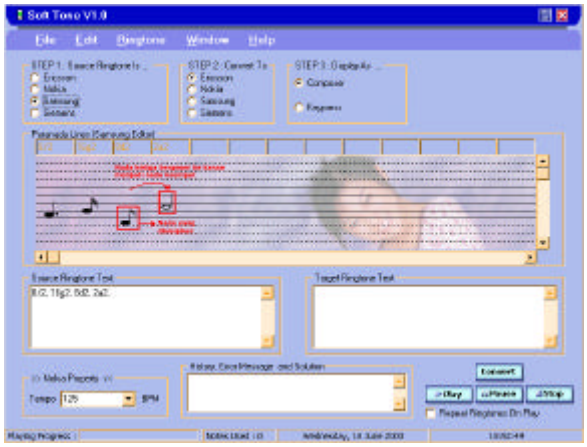


Gambar 10. Hasil Peletakkan Nada Pertama pada Posisi Awal *Editor*

- Untuk proses *editing* menghapus nada, tekan tombol kanan *mouse* pada nada yang akan dihapus maka nada akan secara otomatis bergeser ke kiri. Gambar 4.40 menunjukkan hasil penghapusan nada kedua pada *editor* not balok. Bandingkan gambar 4.40 dengan gambar 4.39 untuk melihat perbedaannya.



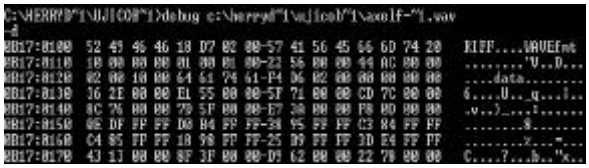
Gambar 11. Hasil Peletakkan Nada Kedua dan Ketiga pada *Editor* Not Balok



Gambar 12. Hasil Penyisipan Nada Baru diantara Nada Kedua dan Nada Ketiga

4.3 Proses Playing dan Pembuatan *Wave File*

Pengujian proses *playing* dan pembuatan *wave file* dilakukan dengan memasukkan *ringtone* pada *editor* dan memasukkan pilihan-pilihan lainnya. Gambar 13 menunjukkan hasil salah satu *wave file* dari *ringtone* axel foley setelah penekanan tombol *play* (dikonversi dari jenis *handphone* Siemens ke Samsung).



Gambar 13. Hasil *Wave File Ringtone* Axel Foley (Dikonversi dari Siemens ke Samsung)

Pada gambar 13 dapat dilihatW bahwa RIFF *chunk* mempunyai panjang total dua belas *byte* (0B17:0100 sampai 0B17:011B). Empat *byte* pertama berisi karakter "RIFF" yang menandakan bahwa *file* tersebut adalah *wave file*, empat *byte* berikutnya menandakan panjang *wave file* (ukuran *wave file* sebenarnya adalah 186,144 *bytes*. Sedangkan panjang *wave file* yang disimpan hanya 186,136 *bytes* karena harus dikurangi dengan empat *bytes* untuk menyimpan karakter "RIFF" dan empat *bytes* untuk menyimpan panjang *wave file*). Sedangkan empat *byte* yang berikutnya berisi karakter "WAVE".

Format chunk mempunyai panjang total duapuluh empat *byte* (0B17:011C sampai 0B17:0123). Empat *byte* pertama berisi karakter "fmt_", empat *byte* berikutnya berisi panjang *format chunk* (selalu 0x00000010 = 16 bytes dalam desimal), 2 *byte* berikutnya berisi audio format (PCM, selalu 0x0001), 2 *byte* berikutnya berisi jumlah *channel* yang digunakan (0x0001 yang berarti jumlah *channel* yang digunakan adalah satu dan *wave file* yang dihasilkan adalah mono), empat *byte* berikutnya berisi *sample rate* (0x00005622 = 22,050 hertz dalam desimal), empat *byte* berikutnya berisi *byte rate* (0x00AC44 menunjukkan bahwa *wave file* tersebut bukan *wave file* 16 bit *stereo*, karena *byte rate* harus empat kali *sample rate* baru *wave file* tersebut dikatakan 16 bit *stereo)*. Dua *byte* berikutnya berisi *bytes per sample* (0x0002 = 2 dalam desimal yang berarti *wave file* adalah 16 bit *mono*) dan 2 *byte* terakhir berisi bits per *sample* (0x0010 = 16 yang menunjukkan bahwa *wave file* tersebut adalah *wave file* 16 bit).

Data *chunk* mempunyai panjang total 8 *byte* ditambah panjang wave data. Empat *byte* pertama berisi karakter "data", empat *byte* berikutnya berisi panjang wave data (0x0002D6F4 = 186,100 bytes) dan *byte* sisanya adalah data (*samples*)

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari penelitian ini dapat diambil beberapa kesimpulan sebagai berikut :

- Tidak semua *ringtone* pada setiap jenis *handphone* dapat dilakukan konversi ke jenis *handphone* yang lainnya karena setiap jenis *handphone* mempunyai *range* nada tertentu.
- *Ringtone* yang berhasil dikonversi dari jenis *handphone* yang satu ke jenis *handphone* yang lainnya, jika dimainkan hasilnya kemungkinan tidak sama dengan *ringtone* aslinya karena konversi *ringtone* bergantung pada *range* nada setiap jenis *handphone* dan proses memainkan *ringtone* bergantung pada tempo masing-masing jenis *handphone.*
- Proses konversi *ringtone* dari jenis *handphone* yang satu ke jenis *handphone* lainnya memperhatikan hal-hal sebagai berikut :
  - *Range* nada yang mencakup ketukan dan oktav dari setiap jenis *handphone*.
  - Pola penulisan *ringtone* yang dimiliki oleh setiap jenis *handphone*.

5.2 Saran

- Program aplikasi ini dapat dikembangkan lebih lanjut dengan menambahkan jenis-jenis *handphone* yang baru seperti Motorola, Philips, LG dan sebagainya, dan mendukung *ringtone polyphonic* dengan menambahkan *multi editor* pada *editor* not balok dan mengubah *format wave file* yg dihasilkan menjadi *file* midi.

DAFTAR PUSTAKA

1. M., DS. Soewito, *Teknik Termudah Bermain Organ I,* Jakarta : Titik Terang, CV, 1992.

2. Sapp, Craig Stuart, *WAVE PCM soundfile format*, http://ccrma-www.stanford.edu/CCRMA/Courses/422/projects/WaveFormat/, 1997.

3. Sentot, *Studi Analisis Format File Suara dan Implementasi Program dengan Sound Blaster*, Surabaya: Teknik Informatika Sekolah Tinggi Teknik Surabaya, 1997.

4. Utdirartatmo, Firrar, *Teori Bahasa dan Otomata,* Yogyakarta: Graha Ilmu J&J Learning, CV, 2000.

5. Csele, Mark S, *COMP630 WAV File Format escription,* http://www. Technology.niagarac.on.ca/courses/comp630/WavFileFormat.html, 2000.

6. Glatt, Jeff, *MIDI Note Number to Frequency Conversion Chart.*, http://www.borg.com/~jglatt/tutr/notefreq.htm, 2002.