ANALISIS PERBANDINGAN KINERJA PROCESSOR DALAM PENGOLAHAN FILE SUARA PADA PERANGKAT ANDROID

Ichwan Kurniawan[1)], Much. Rifqi Maulana[,2)], Arochman Arochman[,3)] Christian Yulianto Rusli[,4)]
STMIK Widya Pratama [1234)]
ichwan.ana10@gmail.com[1)], rifqi@stmik-wp.ac.id[2)], arochman.aryanta@gmail.com[3)]
cyr.tata@gmail.com[4)]

Abstrak

Efiesiensi pekerjaan menjadi hal yang penting pada era digital sekarang ini, salah satunya adalah dalam bidang pengolahan suara. Secara konvisional pengolahan suara menggunakan pengolahan berbasis listrik. Tentunya dengan cara berbasis listrik membutuhkan perangkat yang sangat banyak. Pemproses audio berbasis digital sekarang telah banyak dikembangankan. Teknik manipulasi sinyal suara termasuk didalamnya adalah tiknik mengurangi masking, memastikan keseimbangan antar sumber, dan mengatasi noise, serta pertimbangan artistik, seperti memilih timbre dan tingkat gelombang buatan. Manipulasi sinyal suara digital dimungkinkan dapat melalui perangkat mobile, tujuannya adalah agar pengguna merasa nyaman dalam pekerjaan tersebut dengan memanfaatkan perangkat mobile smart phone. Hasil perbandingan prosentase kinerja *processor* dalam tiga kondisi dapat dilihat bahwa, rata-rata selilih yang paling besar berada pada pengujian perbandingan yang pertama. Karena perbadingan pada saat *app* tidak memainkan musik dengan *app* yang memainkan musik, *processor* memiliki selisih cenderung besar. Sedangkan pada pebandingan kedua kinerja *processor* cenderung stabil, hal ini dikarenakan pengujian perbandingan *app* sama-sama memainkan musik. Kemudian perbadingan ketiga *processor* hanya dibebankan intruksi untuk load data 1 *track* musik, sedangkan pembandingnya ditambakan load data musik dan *effect.* rata kinerja *processor* cenderung turun, hal ini dikarenakan saat *app* memainkan *effect* tidak menggunakan *processor,* namun *app* akan melakukan *load code* ke dalam *memory.*

Kata Kunci : ***Kinerja Processor, Pengolahan Suara, Perangkat Android.***

1. Pendahuluan

Penggunaan sinyal digital audio memungking data dapat dimanipulasi dengan lebih luas, baik dalam manipulasi gelombang maupun durasi sinyal audio (Steinmetz, Pons, Pascual, & Serrà, 2021). Termasuk didalamnya adalah tiknik perekaman dan produksi, baik berupa digital maupun analog yang tentunya melewati beberapa tahapan seperti pengaturan frekuensi, dinamika, ruang dimensi, volume, mengurangi masking, mengatur keseimbangan suara, dan mengtur noise (G.Kharoris, Bintarto, & Satria, 2024). Selain itu Penggunaan metode dalam pemrosesan sinyal audio dapat meningkatkan dan dapat memodifikasi sinyal audio (Sudhamsu & Shastry, 2023). Teknik seperti ini biasanya membutuhkab banyak tools yang digunakan, baik perngkat pemrposes fisik sinyal listrik maupun perangkat pemproses yang berbentuk perangkat komputer.

Melalui perangkat komputer, pemrosesan sinyal audio dapat dilakukan dengan melibatkan logika pemrograman. Selain itu perngkat luna komputer dapat digunakan untuk mengendalikan berbagai paramater efek audi secara *real-time* (Cannon, Fang, & Saniie, 2024). Pemrosesan sinyal audio menjadi hal yang sangat penting dalam semua aplikasi telokomunikasi, dalam pemrosesan sinyal audio perangka multimedia dan audio menjadi hal yang penting (Sudhamsu & Shastry, 2023). Perangkat komputer yang digunakan sebagai pemroses data adalah *Central Processing Unit (CPU)* (Fadilah, Rizky, Sukira, & Aribowo, 2023). Secara digital komputer dapat memproses sinyal audio dengan lebih komplek, hal ini dikarenakan sinyal audio tidak berupada data elektik namu berbentuk data binar yang dapat diolah oleh *processor* secara *real-time.*

Manipulasi sinyal suara digital dimungkinkan dapat melalui perangkat mobile, tujuannya adalah agar pengguna merasa nyaman dalam pekerjaan

tersebut dengan memanfaatkan perangkat *mobile smart phone* (Kang, Chun, Kim, Bo Kim, & Ryong Kim , 2011). Dengan perangkat mobile, pengolahan sinyal suara dapat dilakukan dengan efisien. Hal ini dimungkinkan karena perkembangan perangkat mobile sekarang ini telah memiliki spesifikasi yang baik untuk melakukan pekerjaan lebih komplek. Perkembangan perangkat mobile yang semakin bagus, proses pengolahan dapat dilakukan dengan mudah dan singkat.

Penjelasan di atas menggambarkan bahwa, pemrosesan sinyal audio dapat dilakukan dengan menggunakan perangkat komputer. Namun, dengan perangkat komputer masih memerlukan perangkat yang cukup komplek, diantaranya *spacker, minitor, keyboard* dan *mouse*. Sedangkan dengan menggunakan perangkat komputer yang dalam bentuk perangkat *mobile*, tidak memerlukan perangkat tersebut. Penelitian ini akan dilakukan analisis pengukuran kinerja *processor* perangkat mobile dalam pemrosesan perbedaan *track* sinyal audio yang dimainkan.

Berdasarakan latar belakang maka dapat diambil rumusan masalahnya adalah "Apakah perbedaan *track* sinyal audio yang dimainkan dapat mempengaruhi *processor perform android?".* Sedangkan tujuan dari penelitian ini adalah Analisis *Processor Perform Android* pada Perbedaan *Track* Sinyal Audio. Kemudian manfaat dari penelitian ini adalah memberikan gambaran kepada komposer audio tentang pengolahan *audio* dapat menggunakan perangkat *mobile*.

2. Metode Penelitian

Penelitian ini adalah penelitian experimen dengan metode penelitian sebagai berikut:

a. Penentuan masalah penelitian adalah dengan menggunakan studi leteratur dan studi lapangan.
b. Penentuan *Computing Approach* penelitian ini dipilih studi literatur mengenai Pembuatan aplikasi pengolah musik berbasis *mobile*, hal ini dikarenakan perangkat *mobile* telah meiliki pernakat *input, process* dan *output* dalam satu perangkat.
c. Analisis *Processor Perform Android* pada Perbedaan *Track* Sinyal Audio
d. Evaluasidengan melakukan komparasi dengan data empiris.
e. Pengujian dengan melakukan perbandingan *processor perform adnroid* pada Perbedaan *Track* Sinyal Audio.

3. Hasil dan Pembahasan

3.1 Pengumpulan Data

Populasi Data yang digunakan dalam penelitian ini adalah berhubungan dengan masalah yang diteliti, data tersbut adalah data yang berhubungan dengan nilai prosentase kinerja *processo* pada saat *track* musik dimainkan. Data kinerja *processor* diperoleh dari *debuge profiler* pada aplikasi *android studio*, durasi pengambilan data diambil saat durasi detik ke 10 sampai detik ke 20 dalam pengujian. Kemudian data akan dibandingkan pada saat musik tidak dimainkan, dimainkan 1 track, dimaikan 2 track dan dimainkan 1 track dengan effect. Sound effect yang digunakan dalam pengujian ini adalah *Low Pass, Hight Pass, Flanger, Phaser, Tremolo, Rinmod, Compressor, Distortion, Hall Convolve, Spring Convolve, Telephone Convolve, Warehouse Convolve* dan *Delay.*

3.2 Penentuan *Computing Approach*

Computing Computing approach pada penelitian ini adalah pengolahan sinya audio dengan menggunakan perangkat *mobile*, penggunaan perangkat *mobile* dipilih dengan alasan bahwa perkembangan perangkat *mobile* sekarang ini mampu melakukan pemrosesan yang besar. Spesifikasi yang mememenuhi, dimungkinkan dapat digunakan untuk mengolah sinyal audio dengan efektif, hal ini karena dengan satu perangkat *mobile* telah dilengkapi perangkat *input* sekaligus *output*.

Aplikasi pengolahan data audio dikembangkan dengan menggunakan *software* pengembang berbasis *cloud Construct3, software* ini mudah dan ringan dalam penggunaannya. Kemudian, *sofrware* ini menyediakan *export* kedalam banyak *platform*, diantaranya *platform mobile android*.

Gambar 1 Pengembangan Aplikasi dengan *Construct 3*

Bagian *layout* berinteraksi langsung dengan pengguna melalui aktifitas *input touch* pada area objek pada *layout*, kemudia pada bagian *event sheet* akan merespon apa yang diinputkan oleh pengguna.

3.3 Implementasi

Implentasi pengolahan sinya audio dengan menggunakan perangkat *mobile*, akan membandingkan kinerja *processor* pada perangkat *android* baik pada saat memainkan musik atau tidak dan pada saat memainkan *sound effect* atau tidak.

Pengamatan kinerja *processor* pada perangkat *android*, *Tools* yang digunakan untuk pengukuran kenerja diperoleh dari aplikasi *android studio* dapat dilihat melalui *profiler timeline devices*, *profiler*. Selanjutnya *android studio* akan merespon aktifitas perangkat *android* selama masa *build devices* berjalan. Pengamatan pada *processor* akan dilakukan dalam 4 tahap, yaitu pengataman pada saat aplikasi tidak memainkan musik, memainkan 1 *track* musik, memainkan 2 *track* musik dan memainkan 1 satu *track* musik dengan *effect*. Selain itu durasi pengamatan yang dilakukan hanya sepanjang waktu 11 detik pengamatan, yaitu dari detik 10 sampai dengan detik 20.

Gambar 2 *Kinerja Processor Perangkat Android*

Pengamatan kinerja *processor* dapat didetailkan dengan melihat prosentase penggunaan *processor* dalam menjalankan *App* dan *Others.*

3.4 Pembahasan

Tabel 1. Hasil Pengamatan Kinerja *Processor* dalam menjalankan *app*

CPU (%)		***Stop***	***Play 1***	***Play 1, 2***	***Play 1. Effect***	**Rata-Rata**
Detik ke	**10**	2	10	10	14	9,0
	11	3	11	11	11	9,0
	12	3	13	12	11	9,8
	13	3	11	14	11	9,8
	14	3	11	12	11	9,3
	15	3	13	11	10	9,3
	16	3	11	15	12	10,3
	17	2	12	11	12	9,3
	18	3	11	10	10	8,5
	19	2	12	13	11	9,5
	20	3	11	11	10	8,8
Rata-Rata		**2,7**	**11,5**	**11,8**	**11,2**	**9,3**

Hasil pengamatan kinerja *processor* perangkat *android* dalam menjalankan *app*, terdapat perbedaan prosentase dalam 4 tahap pengamatan dalam waktu 11 detik. Saat *app* tidak memainkan musik rata-rata prosentase *processor* sebesar 2.7 %, memainkan musik 1 track 11,5 %, memainkan musik 2 track 11,8 % dan memainkan 1 track musik dengan effect 11,2 %.

Jika dilihat, prosentase kinerja processor cenderung meningkat pada saat track musik dimainkan, baik dimainkan 1 track maupun 2 track. Rata-rata prosentase kinerja processor paling rendah adalah pada saat app tidak memainkan musik, yaitu 2,7 %. Sedangkan rata-rata prosentase kinerja processor paling besar adalah pada saat app memainkan 2 track musik, yaitu 11,8 %.

3.5 Evaluasi

Tahap pengujian ini akan dilakukan beberapa tahap, dengan menguji kinerja prosentase *processor* pada saat app dijalankan melalui perangkat *android*. Selain itu pengujian kinerja

processor akan dibandingkan dalam tiga kondisi. Pertama, perbandingan pada saat app tidak memainkan musik dengan *app* memainkan musik. Kedua, perbandingan pada saat *app* memainkan 1 *track* musik dengan *app* memainkan 2 *track* musik. Ketiga, perbandingan pada saat *app* memainkan 1 *track* musik dengan *app* memainkan 1 *track* musik dengan *effect*.

Tabel 2. Perbandingan Kinerja *Processor* Kondisi Pertama

CPU (%)		Stop	Play 1	Selisih
Detik ke	10	2	10	8
	11	3	11	8
	12	3	13	10
	13	3	11	8
	14	3	11	8
	15	3	13	10
	16	3	11	8
	17	2	12	10
	18	3	11	8
	19	2	12	10
	20	3	11	8
Rata-Rata		**2,7**	**11,5**	**8,7**

Jika dilihat selisih prosentase kinerja *processor* pertama, yaitu 8% dan 10%. Hal ini dimungkinkan karena pada saat *app* tidak memainkan musik, *processor* belum terbebani oleh intruksi program dengan rata-rata kinerja 2,7 %. Sedangkan pada saat *app* memainkan 1 *track* musik, *processor* telah dibebani oleh intruksi progam dengan rata-rata kenerja 11,5 %. Rata-rata selisih prosentase kinerja *processor* pada kondisi pertama ini adalah sebesar 8,7 %, dengan selisih yang cukup besar.

Tabel 3. Perbandingan Kinerja *Processor* Kondisi Kedua

CPU (%)		Play 1	Play 1, 2	Selisih
Detik ke	10	10	10	0
	11	11	11	0
	12	13	12	1
	13	11	14	3
	14	11	12	1
	15	13	11	2
	16	11	15	4
	17	12	11	1
	18	11	10	1
	19	12	13	1
	20	11	11	0
Rata-Rata		**11,5**	**11,8**	**1,3**

Jika dilihat selisih prosentase kinerja *processor* kedua, yaitu 0% *sampai* dengan 4%. Hal ini dimungkinkan karena pada saat *app* memainkan 1 *track* musik, *processor* hanya terbebani oleh 1 intruksi program untuk load data musik dengan rata-rata kinerja 11,5 %. Sedangkan pada saat *app* memainkan 1 dan 2 *track* musik, *processor* telah dibebani oleh 2 intruksi progam untuk load data musik dengan rata-rata kinerja 11,8 %. Rata-rata selisih prosentase kinerja *processor* pada kondisi kedua ini adalah sebesar 1,3 %, dengan selisih yang cukup rendah.

Tabel 4. Perbandingan Kinerja *Processor* Kondisi Ketiga

CPU (%)		Play 1	Play 1, Effect	Selisih
Detik ke	10	10	14	4
	11	11	11	0
	12	13	11	2
	13	11	11	0
	14	11	11	0
	15	13	10	3
	16	11	12	1
	17	12	12	0
	18	11	10	1
	19	12	11	1
	20	11	10	1
Rata-Rata		**11,5**	**11,2**	**1,2**

Jika dilihat selisih prosentase kinerja *processor* ketiga, yaitu 0% sampai dengan 4%. Hal ini dimungkinkan karena pada saat *app* memainkan 1 *track* musik tanpa *effect*, *processor* hanya terbebani oleh 1 intruksi program untuk load data musik dengan rata-rata kinerja 11,5 %. Sedangkan pada saat *app* memainkan 1 *track* musik dengan *effect*, *processor* telah dibebani

oleh 1 intruksi progam untuk load data musik dengan rata-rata kinerja 11,2 %. Rata-rata selisih prosentase kinerja *processor* pada kondisi ketiga ini adalah sebesar 1,2 %, dengan selisih yang cukup rendah. Namun, jika diperhatikan pada saat memainkan musik dengan *effect* rata-rata kinerja *processor* cenderung turun, hal ini dikarenakan saat *app* memainkan *effect* tidak menggunakan *processor*, namun *app* akan melakukan *load code* ke dalam *memory*.

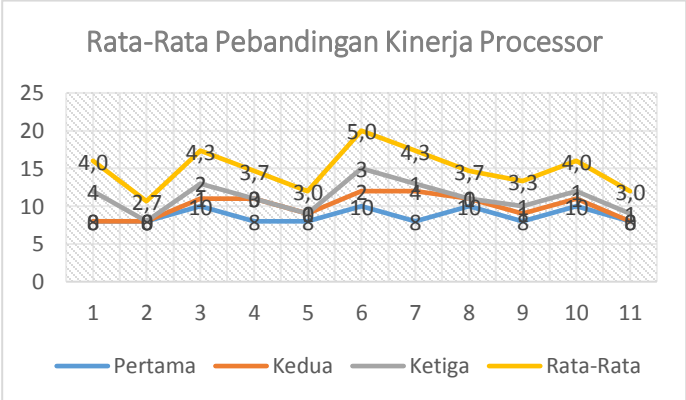
Tabel 5. Rata-Rata Perbandingan Kinerja *Processor*

CPU (%)		1	2	3	Rata-Rata
Detik ke	10	8	0	4	4,0
	11	8	0	0	2,7
	12	10	1	2	4,3
	13	8	3	0	3,7
	14	8	1	0	3,0
	15	10	2	3	5,0
	16	8	4	1	4,3
	17	10	1	0	3,7
	18	8	1	1	3,3
	19	10	1	1	4,0
	20	8	0	1	3,0
Rata-Rata		**8,7**	**1,3**	**1,2**	**3,7**

Hasil perbandingan prosentase kinerja *processor* dalam tiga kondisi dapat dilihat bahwa, kondisi pertama yaitu perbandingan pada saat *app* tidak memainkan musik dengan *app* memainkan musik menghasilkan rata-rata prosentase kinerja *processor* sebesar 8,7 %. Kedua, perbandingan pada saat *app* memainkan 1 *track* musik dengan *app* memainkan 2 *track* musik, menghasilkan rata-rata prosentase kinerja *processor* sebesar 1,3 %. Ketiga, perbandingan pada saat *app* memainkan 1 *track* musik dengan *app* memainkan 1 *track* musik dengan *effect*, menghasilkan rata-rata prosentase kinerja *processor* sebesar 1,2 %. Sedangkan rata-rata dari rata-rata kinerja tiap detik pengukurannya adalah sebesar 3,7 %.

Hasil penyujian perbandingan kinerja *processor* dalam tiga kondisi dapat disimpulkan bahwa, rata-rata selilih yang paling besar berada pada pengujian perbandingan yang pertama, yaitu pada pengujian perbandingan pada saat *app* tidak memainkan musik dengan *app* memainkan musik. Hali ini, disebkan karena pada saat *app* tidak memainkan musik, *processor* belum ada beban inturksi/pekarjaa, sedangkan pada saat *app* memainkan musik, *processor* sudah ada beban intukris untuk load data musik untuk dimainkan. Sedangkan hasil penyujian perbandingan kinerja *processor* yang kedua dan ketiga cenderung hampir sama, selisih 0,1 %.

Hal ini dikarenakan bahwa, pada pengujian yang kedua *processor* sama-sama telah dibebankan intruksi untuk load data musik. Pengujian kedua, perbandingan pada saat *app* memainkan 1 *track* musik dengan *app* memainkan 2 *track* musik. Dalam hal ini, *processor* sama-sama melakukan load data music, pembanding dengan 1 *track* musik dan yang dibandingkan dengan 2 *track* musik. Perbadingan beban *processor* turun 7,5 %. Sedangkan pada pengujian yang ketiga, perbandingan pada saat *app* memainkan 1 *track* musik dengan *app* memainkan 1 *track* musik dengan *effect*. Pengujian ketiga cenderung turun dibandingkan dengan pengujian kedua, hal ini dikarena pada pengujian kedua *processor* mempunyai beban intruksi yang besar, karena harus load data musik, yaitu 1 *track* musik dan 1, 2 *track* musik. sedangkan pada pengujian yang ketiga *processor* hanya dibebankan intruksi untuk load data 1 *track* musik, sedangkan pembandingnya ditambakan load data musik dan *effect*. rata kinerja *processor* cenderung turun, hal ini dikarenakan saat *app* memainkan *effect* tidak menggunakan *processor*, namun *app* akan melakukan *load code* ke dalam *memory*.



Gambar 3 Rata-Rata Prosentase Kinerja *Processor*

4. Kesimpulan dan Saran

Hasil perbandingan prosentase kinerja *processor* dalam tiga kondisi dapat dilihat bahwa, kondisi pertama yaitu perbandingan pada saat *app* tidak memainkan musik dengan *app* memainkan musik menghasilkan rata-rata prosentase kinerja *processor* sebesar 8,7 %. Kedua, perbandingan pada saat *app* memainkan 1 *track* musik dengan *app* memainkan 2 *track* musik, menghasilkan rata-rata prosentase kinerja *processor* sebesar 1,3 %. Ketiga, perbandingan pada saat *app* memainkan 1 *track* musik dengan *app* memainkan 1 *track* musik dengan effect, menghasilkan rata-rata prosentase kinerja *processor* sebesar 1,2 %. Sedangkan rata-rata dari rata-rata kinerja tiap detik pengukurannya adalah sebesar 3,7 %.

Hasil penyujian perbandingan kinerja *processor* dalam tiga kondisi dapat disimpulkan bahwa, rata-rata selilih yang paling besar berada pada pengujian perbandingan yang pertama, yaitu pada pengujian perbandingan pada saat *app* tidak memainkan musik dengan *app* memainkan musik. Hali ini, disebkan karena pada saat *app* tidak memainkan musik, *processor* belum ada beban inturksi/pekarjaa, sedangkan pada saat *app* memainkan musik, *processor* sudah ada beban intukris untuk load data musik untuk dimainkan. Sedangkan hasil penyujian perbandingan kinerja *processor* yang kedua dan ketiga cenderung hampir sama, selisih 0,1 %.

Hal ini dikarenakan bahwa, pada pengujian yang kedua *processor* sama-sama telah dibebankan intruksi untuk load data musik. Pengujian kedua, perbandingan pada saat *app* memainkan 1 *track* musik dengan *app* memainkan 2 *track* musik. Dalam hal ini, *processor* sama-sama melakukan load data music, pembanding dengan 1 *track* musik dan yang dibandingkan dengan 2 *track* musik. Perbadingan beban *processor* turun 7,5 %. Sedangkan pada pengujian yang ketiga, perbandingan pada saat *app* memainkan 1 *track* musik dengan *app* memainkan 1 *track* musik dengan *effect.* Pengujian ketiga cenderung turun dibandingkan dengan pengujian kedua, hal ini dikarena pada pengujian kedua *processor* mempunyai beban intruksi yang besar, karena harus load data musik, yaitu 1 *track* musik dan 1, 2 *track* musik. sedangkan pada pengujian yang ketiga *processor* hanya dibebankan intruksi untuk load data 1 *track* musik, sedangkan pembandingnya ditambakan load data musik dan *effect.* rata kinerja *processor* cenderung turun, hal ini dikarenakan saat *app* memainkan *effect* tidak menggunakan *processor*, namun *app* akan melakukan *load code* ke dalam *memory.*

Daftar Pustaka

Cannon, D., Fang, T., & Saniie, J. (2024). Modular Delay Audio Effect System on FPGA. *2022 IEEE International Conference on Electro Information Technology (eIT).* Mankato, MN, USA: IEEE. doi:https://doi.org/10.1109/eIT53891.2022.9813875

Fadilah, S. A., Rizky, M., Sukira, S., & Aribowo, D. (2023, 12 4). Mengevaluasi Efisiensi Pengontrol Input-Output dalam Arsitektur Komputer Modern. *Jurnal Teknik Mesin, Industri, Elektro Dan Informatika (JTMEI)*, 96-113. doi:https://doi.org/10.55606/jtmei.v2i4.2982

G.Kharoris, S., Bintarto, G., & Satria, E. (2024). Analisis Proses Mixing Vokal pada Lagu Daddy's Fav Boy di Saga Audio Music Production. *IDEA: Jurnal Ilmiah Seni Pertunjukan*, 133 - 145. Diambil kembali dari https://journal.isi.ac.id/index.php/IDEA/article/view/9103/3547

Kang, J. A., Chun, C., Kim, H., Bo Kim, M., & Ryong Kim , S. (2011). A smart background music mixing algorithm for portable digital imaging devices. *IEEE Transactions on Consumer Electronics*, 1258 - 1263. doi:https://doi.org/10.1109/TCE.2011.6018882

Steinmetz, C. J., Pons, J., Pascual, S., & Serrà, J. (2021). Automatic Multitrack Mixing With A Differentiable Mixing Console Of Neural Audio Effects. *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (hal. 71 - 75). Toronto, ON, Canada: IEEE. doi:https://doi.org/10.1109/ICASSP39728.2021.9414364

Sudhamsu, G., & Shastry, B. (2023). Audio Signal Processing Using MATLab. *2023 International Conference on Network, Multimedia and Information Technology (NMITCON).* Bengaluru, India: IEEE. doi:https://doi.org/10.1109/NMITCON58196.2023.10276228